JRC Scientific and Technical Reports

Optimization Techniques for MIMO radar antenna systems

Elias Mendez-Dominguez
Joaquim Fortuny-Guasch

EUR 23671 EN - 2008

JRC
EUROPEAN COMMISSION

ipsc
Institute for the Protection and Security of the Citizen

The Institute for the Protection and Security of the Citizen provides research based, systems-oriented support to EU policies so as to protect the citizen against economic and technological risk. The Institute maintains and develops its expertise and networks in information, communication, space and engineering technologies in support of its mission. The strong crossfertilisation between its nuclear and non-nuclear activities strengthens the expertise it can bring to the benefit of customers in both domains.

European Commission
Joint Research Centre
Institute for the Protection and Security of the Citizen

Contact information
Address: Centro Comune di Ricerca
Via Enrico Fermi 1, Ispra (VA) 21020 ITALY

E-mail: elias.mendez@jrc.it
Tel.: +39 0332 789020
Fax: +39 0332 785469

http://www.jrc.ec.europa.eu

JRC 49366
EUR 23671 EN
ISSN 1018-5593

Luxembourg: Office for Official Publications of the European Communities

Printed in Italy

Optimization Techniques for MIMO radar antenna systems

Elias Mendez-Dominguez
Joaquim Fortuny-Guasch

December 2008

Contents

1. Introduction: Topology Optimization Technique.

A new technique to obtain optimum topologies for MIMO antenna array systems has been developed. The target of the optimization is the identification of the optimal arrangement of the transmitters and receivers giving the highest detection performance to obtain radar images as similar as possible to those obtained in SAR techniques. To carry out this task we will focus on the concept of the *phase center*.

In navigation, tracking, homing, landing, and other aircraft and aerospace systems it is usually desirable to assign to the antenna system a reference point such that for a given frequency, the radiation pattern is independent of the angles θ and φ (azimuth and elevation respectively). When referenced to the phase center, the fields radiated by the antenna are spherical waves with ideal spherical wave fronts or equiphase surfaces. Therefore a phase center is a reference point from which radiation is said to emanate, and radiated fields measured on the surface of a sphere whose center coincides with the phase center have the same phase [2].

To calculate the phase centers (PC) of an array antenna, we proceed as follows:

$$PC(i,j) = \frac{Tx(i) + Rx(j)}{2} \qquad \forall i,j / \quad i = 1,2,3,...,NantT \qquad j = 1,2,3,...,NantR \tag{1}$$

Tx(i) and *Rx(j)* are the x-coordinates of the transmitter *i* and receiver *j* respectively.

If the antenna topology is not well designed, the appearance of multiple repeated phase centers can occur (see Figure 2). To solve this problem so that only singleton phase centers appear (see Figure 3), the one dimensional topology will be studied. Since in the one dimensional topology the transmitters T_x and receivers R_x are placed in the x-axis, T_x and R_x can be written as vectors/arrays in the x coordinates space. To improve the quality of the image processed by the radar, uniformity (equidistant) in the phase centers locations is also required. Topologies with an unbalanced number of transmitters and receivers are not considered because of the complexity of the antenna feeding network.

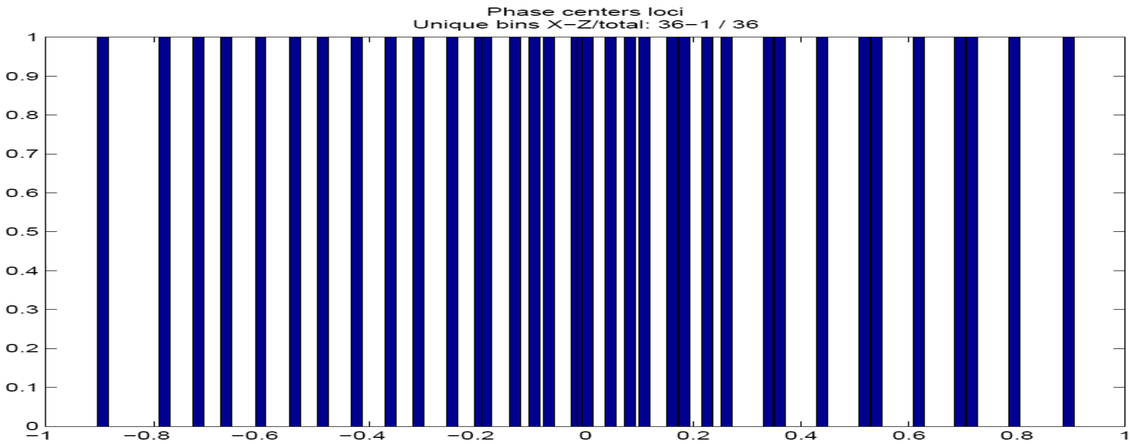


Figure 1.- Non uniform phase centers.

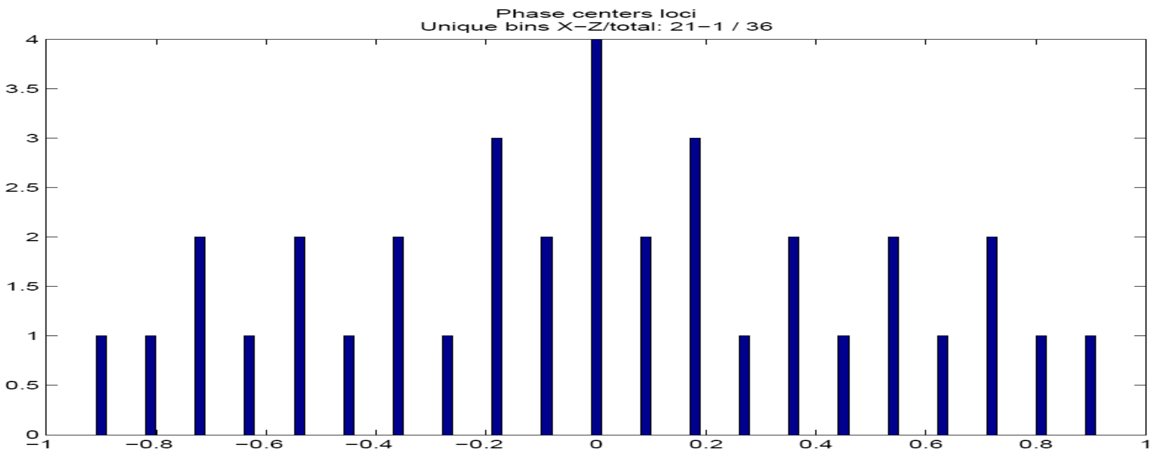


Figure 2.- Repeated phase centers.

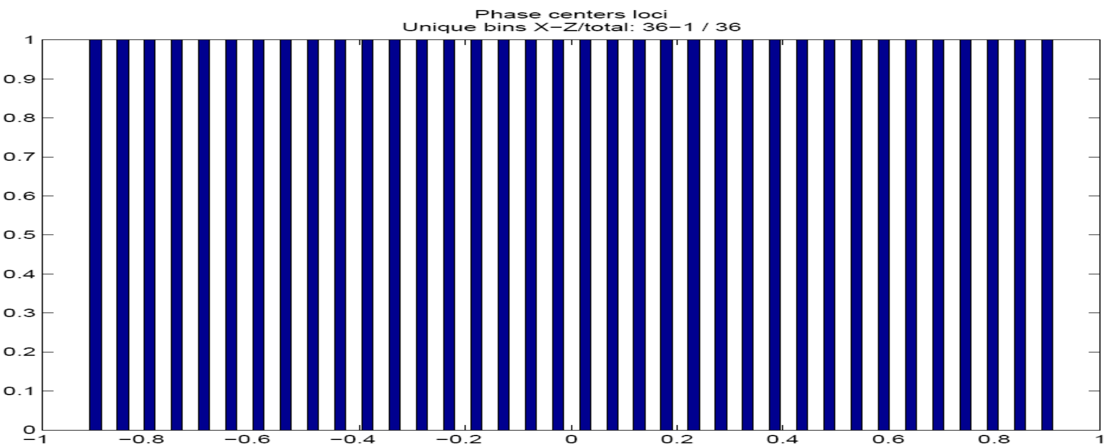


Figure 3.- Uniform and unique phase centers.

Moreover another condition will be imposed to increase the efficiency of the antenna. Considering L_e as the length of the physical aperture of the antenna and L the length of the phase centers associated, the topology should fulfill $\Delta_{Le} \geq |L - L_e|$. In Figure 4, it can be seen the difference between $|L - L_e|$ for the first and the second arrangement of T_x and R_x (Figures 4.a and 4.b respectively). The first topology will not be considered optimum because the phase centers do not cover all the physical aperture of the antenna.

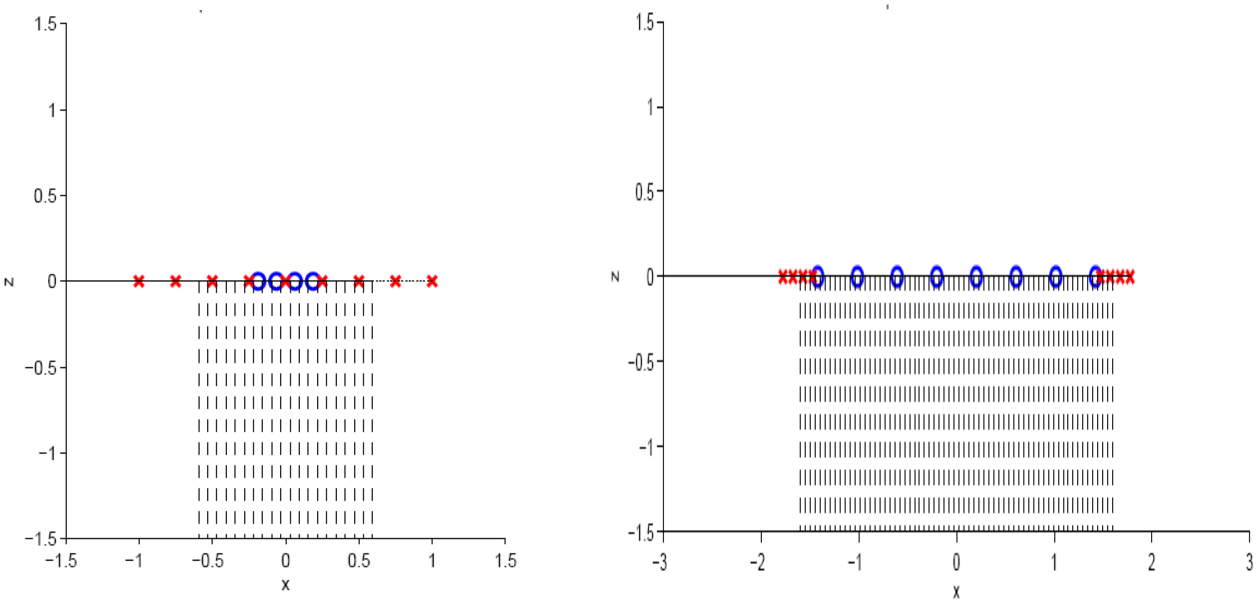


Figure 4.- T_x - R_x antennas and associated phase centers.
4.a Short aperture extent. 4.b. Large aperture extent.

This condition can be expressed with the following equation:

$$\Delta L_e \leq |L - L_e| = sum(diff(sort(cat(2, Tx, Rx)))) - sum(diff(sort(PC))) \quad (2)$$

The ratio L_e/L will allow us to discard topologies when brute force algorithms are applied. When these methods are used, a big set of optimum pairs T_x/R_x can be given as a result, in that case L_e/L must be maximize to obtain the best solution to the optimization problem.

Another characteristic the topology should fulfill is the free alias region condition. To fulfill this condition it is necessary that:

$$\frac{\lambda}{4} = \Big|_{5GhZ} 0.015 \leq \Delta_{Uniformity} \tag{3}$$

Taking all these points into account, the main routine (programmed in a m-function) to calculate and optimum antenna topology will proceed as follows:

1. - Given the desired number of the different phase centers positions 'NPC', the different combinations of possible number of transmitters and receivers are calculated to fix Nt and Nr so that Nt*Nr=NPC. Among all these possible combinations between transmitters and receivers, the user can select the desired one to study and optimize.

2. - According to the selected combination (number of transmitters and receivers), the user specifies the algorithm to carry out the optimization: Non Constrained Linear Least Squares, Constrained Linear Squares, Non Linear Least Squares, Genetic Algorithm, Brute Force and Direct Search. Notice that there are more than one variant for all these techniques due to the fact that they consider different schemes for the vectors T_x and/or R_x.

In this step we will introduce the concept of optimization in the mathematical formalism so that the reader can understand the contents of this document in a proper way.

Given a function $f : R^n \to R,$ and a set $S \subseteq R^n$, "optimization" means finding $x^* \in S$ such that $f(x^*) \leq f(x)$ for all $x \in S$. The points $x \in S$ are called *feasible* points and the point x^* is called the *minimizer* or *minimum* of the function *f*. Its enough to consider only minimization, since maximum of *f* is minimum of -*f*. The *objective* function *f* is usually assumed differentiable, and may be linear or nonlinear. There might also be a *constraint* set *S* defined by a system of equations and equalities that may be linear of nonlinear. If $S = R^n$, the problem is *unconstrained* (min *f(x)* for all $x \in R^n$). If the problem is linearly constraint, then min *f(x)* should be found for all *x* such that $Ax \leq b$ and/or $A_{eq}x = b_{eq}$; where *A* and A_{eq} are matrices and *b* and b_{eq} are vectors. The possibility of considering the constraint $LB \leq x \leq UB$ will be also foreseen, where *LB* and *UB* will be vectors or real numbers.

$x^* \in S$ is a global minimum if $f(x^*) \leq f(x)$ for all $x \in S$.

$x^* \in S$ is a local minimum if $f(x^*) \leq f(x)$ for all feasible x in some neighborhood of x^*.

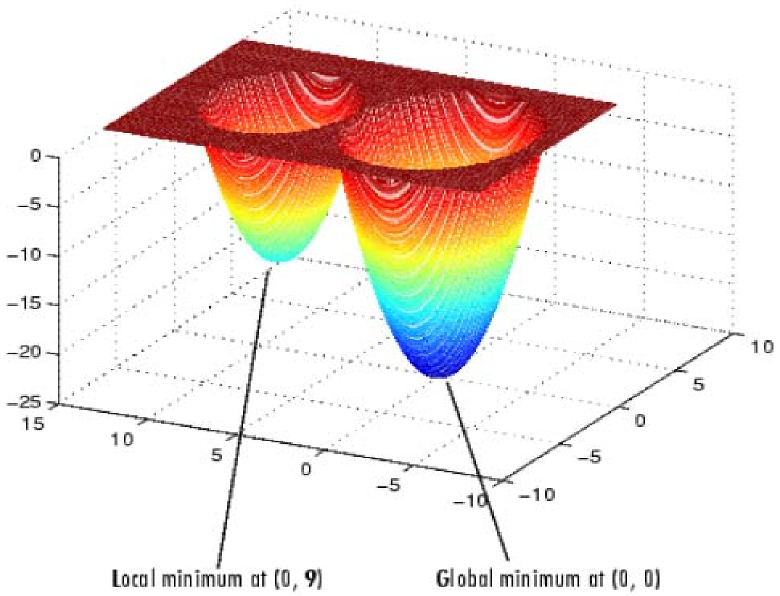


Figure 5.- Local minimum versus Global minimum.

Finding, or even verifying global minimum is difficult, in general. Most of the optimization methods are designed to find local minimum, which may or may not be global minimum. If a global minimum is desired, we can try several widely separated starting points and see if all of them produce the same results. For some problems, such as linear programming, global optimization is tractable.

1.1. Brute Force Algorithms (BF).

The Brute Force algorithm is a trivial but very general problem solving technique, which consists of systematically enumerating all possible candidates for the solution and checking whether each candidate satisfies the problem's statement. Brute Force search is simple to implement, and will always find a solution if it exists. However, its cost is proportional to the number of candidate solutions, which, in many practical problems, tends to grow very quickly as the size of the problem increases. Therefore, brute force search is typically used when the problem size is limited, or when there are problem specific heuristics that can be used to reduce the set of candidate solutions to a manageable size. The method is also used when the simplicity of implementation is more important than speed. In our case the BF algorithm will find optimum antenna topologies for Nt transmitters and Nr receivers, which can be placed in a one dimensional vector. This vector is refereed as 'vectorPos' with a length of 'xPos = length (vectorPos)' different values and obtained as 'vectorPos = - Cte_1: resolution: Cte_1'. Typically, resolution = 0.0001 and Cte_1 = -1.2.

As the BF algorithm has to calculate all the possible candidate solutions for Nt and Nr transmitters and receivers respectively, the number of operations (or computational cost) is:

$$NC = \binom{xPos}{Nt}\binom{xPos}{Nr}$$

$$NC = \frac{factorial(xPos)}{factorial(xPos - Nt) * factorial(Nt)} * \frac{factorial(xPos)}{factorial(xPos - Nr) * factorial(Nr)} \tag{4}$$

Raising the problem in that way requires an extremely high computational cost and subsequently runtimes in the range of weeks, months or even much higher. In a first attempt to reduce the runtime and considering the general case for any value of Nr and Nt, the Brute Force algorithm will proceed in the following way:

- ❖ An equidistant vector for R_x is calculated by applying the m-function 'linspace', for example: R_x=linspace (K_1, K_2, Nr), where typically K_1=K_2=1.
- ❖ The one dimensional vector 'vectorPos' with all the possible locations of the Nt transmitters is calculated. Notice also that the ***resolution*** of this vector will be reflected in the runtime of the algorithm.
- ❖ The transmitters are placed in 'vectorPos' without superimposing and repeating. Then all the possible combinations for the 'Nt' transmitters placed in all the 'length(vectorPos)' possible positions are evaluated.

The drawback of this method lies on the number of candidate solutions that need to be checked, thus in this situation is:

$$NC = \binom{length(vectorPos)}{Nt} = \frac{factorial(length(vectorPos))}{factorial(length(vectorPos) - Nt) * factorial(Nt)} \tag{5}$$

If the symmetrical scenario can be considered (and therefore Nt must be an even number), then the number of operations to be done can considerably be reduced compared with those needed to do in the previous situation. The possible T_x vectors can be expressed as:

T_x = [-x(1) -x(2) -x(3) … -x(Nt/2)] U [x(1) x(2) x(3) … x(Nt/2)] (6)

In these topologies, the number of variables to be calculated is Nt/2 decreasing in this way the runtime of the algorithm. Other alternatives consist of applying a 'linspace' function to the transmitters when the receivers are calculated as an equidistant vector too.

Let us consider the following configurations for the pairs T_x/R_x:

```
Tx1 = cat (2, linspace (-A, -B, Nt/2), linspace (B, A, Nt/2))

Rx1 = linspace (-C, C, Nr)
```

```
Tx2 = sort (cat(2,linspace(C-B/2, C+B/2, Nt/2), - linspace(C-B/2, C+B/2, Nt/2)))

Rx2 = linspace (-A/2, A/2, Nr)
```

With these configurations the maximum number of variables needed to calculate with the BF algorithm is 3 (A, B and C). Figure 6 shows these configurations with the main parameters involved. In Figure 6.a the pair Tx1/Rx1 is represented, but it is also possible to swap Tx1 with Rx1 and vice versa. Δt and Δr are the value of the distance between consecutive transmitters or receivers respectively. Figure 6.b represents the configuration for the pair Tx2/Rx2.

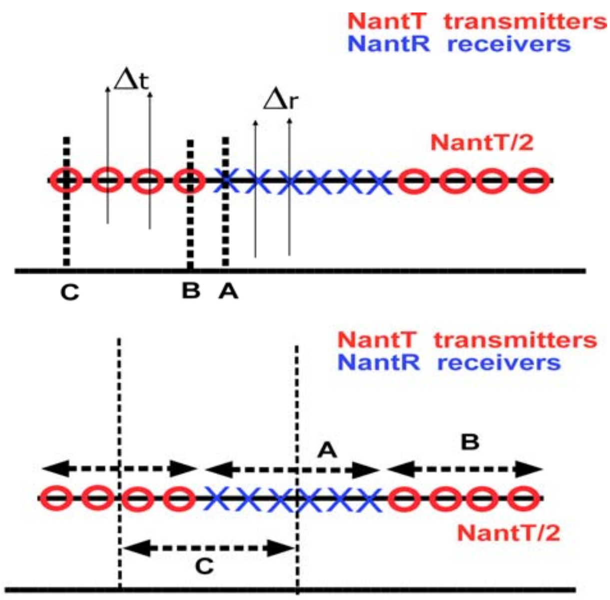


Figure 6.- Schemes for the topologies.
6.a. Topology for Tx1 and Rx1. 6.b. Topology for Tx3 and Rx3.

These algorithms will give all the possible solutions for a topology with Nt transmitters and Nr receivers. These topologies will fulfill uniqueness and uniformity condition but the value $|L - L_e|$ must be calculated afterwards to obtain the efficiency of the topology according to the positions of the phase centers. The results obtained with these algorithms will be presented in section 2.4.

1.2. Linear Least Squares (LLS) Method.

To optimize the topology, both constrained and non constrained LLS methods are explained. These methods try to find an optimum solution for a given vector (x-coordinates of transmitters and receivers) in the least squares sense. By using the m-function "lsqlin", Matlab calculates (for both variants) the optimum solution according to the expression:

$$\min_{x} \frac{1}{2} \| Cx - d \|_2^2 \tag{7}$$

The constrained variant should proceed taking the equations (4) into account:

$$Ax \le b \ , \ A_{eq} x = b_{eq} \ , \ LB \le x \le UB \tag{8}$$

If the non constrained variant is selected, the system needed to apply in the m-function is calculated in the way $Cx=d$ (e.g. for 3 transmitters and 4 receivers):

$$\left(\begin{array}{ccc|cccc}
0.5 & 0 & 0 & 0.5 & 0 & 0 & 0 \\
0.5 & 0 & 0 & 0 & 0.5 & 0 & 0 \\
0.5 & 0 & 0 & 0 & 0 & 0.5 & 0 \\
0.5 & 0 & 0 & 0 & 0 & 0 & 0.5 \\
\hline
0 & 0.5 & 0 & 0.5 & 0 & 0 & 0 \\
0 & 0.5 & 0 & 0 & 0.5 & 0 & 0 \\
0 & 0.5 & 0 & 0 & 0 & 0.5 & 0 \\
0 & 0.5 & 0 & 0 & 0 & 0 & 0.5 \\
\hline
0 & 0 & 0.5 & 0.5 & 0 & 0 & 0 \\
0 & 0 & 0.5 & 0 & 0.5 & 0 & 0 \\
0 & 0 & 0.5 & 0 & 0 & 0.5 & 0 \\
0 & 0 & 0.5 & 0 & 0 & 0 & 0.5
\end{array}\right)
\begin{pmatrix} x(1) \\ x(2) \\ x(3) \\ x(4) \\ x(5) \\ x(6) \\ x(7) \end{pmatrix}
=
\begin{pmatrix} PC(1) \\ PC(2) \\ PC(3) \\ PC(4) \\ PC(5) \\ PC(6) \\ PC(7) \\ PC(8) \\ PC(9) \\ PC(10) \\ PC(11) \\ PC(12) \end{pmatrix} \tag{9}$$

The simple '*C* \ *d*' option is valid to calculate the solution *x* of the system. Nevertheless the algorithm will find a global, although not necessarily unique, solution; indeed the solution may not fulfill the condition $\Delta_{Le} \geq |L - L_e|$. If the '*C* \ *d*' is chosen, then the matrix *d* containing the positions of the phase centers must be provided. The Matlab command for both constrained and non-constrained methods can be resumed in the expression:

$$\text{[X,RESNORM,RESIDUAL,EXITFLAG,OUTPUT]=LSQLIN(C,d,A,b,}A_{eq}\text{,}b_{eq}\text{,LB,UB,}X_0\text{,OPTIONS)} \quad (10)$$

- X is the solution of the method.
- RESNORM: returns the value of the squared 2-norm of the residual, norm(C*x-d)^2.
- RESIDUAL: returns the residual C*x-d.
- EXITFLAG: returns a value that describes the exit condition of the algorithm.
  - '1': The function converged to a solution X.
  - '3': Change in the residual was smaller than the specified tolerance.
  - '0': Number of iterations exceeded options.MaxIter.
  - '-2': The problem is infeasible.
  - '-4': Ill-conditioning prevents further optimization.
  - '-7': Magnitude of search direction became too small. No further progress could be made.
- OUTPUT: returns a structure output that contains information about the optimization. The most important fields are the followings:
  - -'iterations': Number of iterations taken to reach the solution X.
  - -'algorithm': Algorithm used in the optimization process.

X_0 is the starting point for the optimization algorithm and OPTIONS is a structure that permits to minimize with the optimization options specified in the structure options (use '*optimset*' to set these options). For the non-constrained variant set A=[ ], b=[ ], A_{eq}=[ ], b_{eq}=[ ], LB=[ ] and UB=[ ].

By using LLS methods, better numerical results are likely if the user specifies equalities explicitly, using A_{eq} and b_{eq} or implicitly, using *LB* and *UB*. To achieve good results the constrained variant should be used. The expression for the system is the same written in equation 9; the difficult part lies on finding the right expressions for A_{eq}, B_{eq}, *LB*, and *UB* for fulfilling the condition $\Delta_{Le} \geq |L - L_e|$. The discussion of this point will not be commented in this document. Moreover due to the difficulty of finding an optimum topology with both non constrained and constrained LLS methods and in case no valid solution is found, a brute force algorithm will proceed to check all possible combinations according to Nt and Nr as well as the number of possible positions in the x-axis. In this way, the shortest vector T_x or R_x will be scaled and scrolled along the x-axis while the longest one will remain unchanged. The longest vector is the same calculated with the LLS methods (although the user can change the code to use another one). It should be necessary to create a new m-file every time we run the program with a different topology because the number of for-loops depends on Nt and Nr. To avoid this situation an auxiliary m-function will write an m-file containing another m-function according to the required number of for-loops and characteristics of the topology under study (with a determined number of transmitters and receivers); otherwise the m-file cannot be used for different topologies. Therefore the number of calculations the method has to carry out is:

$$NC = \binom{xPos}{N_{t/r}} = \frac{factorial(xPos)}{factorial(xPos - N_{t/r}) * factorial(N_{t/r})} \tag{11}$$

In equation 11 *xPos* are the possible positions in the x-axis the transmitters/receivers can have; $N_{t/r}$ is the number of transmitters/receivers needed to scroll along x-axis. This method will be studied in depth in the next section. Even though this method did not reach a valid solution, then a brute force method (for Nt and Nr variables) would start computing all the possible combinations for Nt, Nr and all the positions *xPos*.

1.3. Non Linear Least Squares Method (NLLS).

The LLS methods have the advantage they can provide an optimum topology (fulfilling uniqueness and uniformity) in really short times (less than 30 seconds) for all different combinations of elements (with equal or different number of transmitters and receivers). These methods provide good solutions if $\Delta_{Le} \geq |L - L_e|$ is not needed to be met. If $\Delta_{Le} \geq |L - L_e|$ is imposed then another technique must be designed to proceed in different way. By supposing symmetry (respect to Y axis) this condition can be easily met. The procedure is the following:

1.- An m-file is created containing the equation written below with the different variables according to the number of transmitters and receivers:

T_x = [x(1) x(3) x(5)…x(2*Nt-1)]; R_x = [x(2) x(4) x(6) …x(2*Nr)] (12)

$$F=\left(\begin{array}{ccc|ccc|ccc}
0.5x(1) & 0.5x(2) & -PC(1) & 0.5x(1) & 0.5x(4) & -PC(2) & 0.5x(1) & 0.5x(2*Nr) & -PC(Nr)\\
\hline
0.5x(3) & 0.5x(2) & -PC(Nr+1) & 0.5x(3) & 0.5x(4) & -PC(Nr+2) & 0.5x(3) & . & -PC(2*Nr)\\
\hdashline
0.5x(5) & 0.5x(2) & \ldots & 0.5x(5) & 0.5x(4) & \ldots & .. & . & -PC(3*Nr)\\
0.5x(7) & 0.5x(2) & \ldots & \ldots & . & \ldots & \ldots & . & \ldots\\
\ldots & . & \ldots & \ldots & . & \ldots & \ldots & . & \ldots\\
\ldots & . & \ldots & \ldots & . & \ldots & \ldots & . & \ldots\\
0.5x(2*Nt-5) & . & \ldots & \ldots & . & \ldots & \ldots & . & -PC((Nt-2)*Nr)\\
0.5x(2*Nt-3) & . & \ldots & \ldots & . & \ldots & \ldots & . & -PC((Nt-1)*Nr)\\
0.5x(2*Nt-1) & 0.5x(2) & -PC((Nt-1)*Nr+1) & .. & . & \ldots & 0.5x(2*Nt-1) & 0.5x(2*Nr) & -PC(Nt*Nr)
\end{array}\right)_{Nt\times Nr}$$

2. - By using the m-function "fsolve" (solve systems of non linear equations), a first approximation of T_x and R_x is obtained. In this part *x* is obtained so that *F(x)=0*, where *F* is the system made up of the equations written in the matrix above. Notice that a non linear least squares method will be applied even when the system is made up of linear equations. The Matlab command can be resumed in the expression:

[xx,fval,exitflag,output,jacobian] = fsolve(fun,x_0,options) **(13)**

- xx is the solution of the method.
- fval: returns the value of the objective function 'fun' at the solution xx.
- exitflag: returns a value that describes the exit condition of the algorithm.
    - '1': The function converged to a solution xx.
    - '2': Change in x was smaller than the specified tolerance.
    - '3': Change in the residual was smaller than the specified tolerance.
    - '4': Magnitude of search direction was smaller than the specified tolerance.
    - '0': Number of iterations exceeded options.MaxIter or number of function evaluations exceeded options.FunEvals.
    - '-1': Algorithm was terminated by the output function.
    - '-2': Algorithm appears to be converging to a point that is not a root.
    - '-3': Trust radius became too small.
    - '-4': Line search cannot sufficiently decrease the residual along the current search direction.
- output: returns a structure output that contains information about the optimization. The most important fields are the followings:
    - -'iterations': Number of iterations taken to reach the solution xx.
    - -'funcCountNumber': Counter of the function evaluations.
    - -'algorithm': Algorithm used in the optimization process.
- jacobian: returns the Jacobian of 'fun' at the solution xx.

x_0 is the starting point for the optimization algorithm and OPTIONS is a structure that permits to minimize with the optimization options specified in the structure options (use '*optimset*' to set these options). These method will provide a solution only if Nt=Nr, otherwise the method will take a extremely high runtime to obtain a solution.

3. - A first evaluation of the topology obtained from the previous step is made. The m-function detects if an optimum topology has already been found (fulfilling uniqueness and uniformity). The antenna aperture length 'L', and the antenna effective aperture length 'L_e' (considering the phase centers positions) are calculated. Finally the condition $\Delta_{Le} \geq |L - L_e|$ is checked. If an optimum topology has been found, then the algorithm halts and the solution is given, otherwise the next step will follow.

4. – A brute force algorithm will proceed due to the fact that an optimum topology has not been found jet. Nevertheless one of the vectors T_x or R_x will remained unchanged with the value calculated with the NLLS method. The brute force algorithm is implemented with four control for-loops. The control for-loops variables *C*, *D*, *E* and *F* are initialized. The m-function calculates the length of the vectors T_x and R_x obtained with the NLLS method, then the minimum between these two lengths is obtained as 'TR'. If TR=0 T_x is the shortest vector, otherwise TR=1. Depending on the value of TR, T_x or R_x will be modified. The variable *D* is calculated as follows:

$$D = \max\left\langle \frac{|\min(T_x)|}{|\min(R_x)|} + \frac{\Delta_{Le}}{2} + 0.2 \quad , \quad \frac{|\min(R_x)|}{|\min(T_x)|} + \frac{\Delta_{Le}}{2} + 0.2 \right\rangle \tag{14}$$

The variables *C* and *E* are initialized to 0.5 and 0.1 respectively. *F* is obtained as:

$$F = |\max(T_x) - \max(R_x)| + \frac{\Delta_{Le}}{2} + 0.4 \tag{15}$$

5. - Expansion technique: According to TR, the vectors obtained with "fsolve" are improved in order to minimize Δ_{Le}. With this method the vector is multiplied by a factor *ind33* (*C* and *D* are the limits in this for-loop) to increase the distance among the transmitters/receivers; in this way Δ_{Le} will decrease too. This technique will give an optimized topology if the uniformity, uniqueness and $\Delta_{Le} \leq \varepsilon$ conditions meet; otherwise the next technique will proceed.

6. - Scrolling technique: According to the variable TR, the vector obtained with "fsolve" is improved in order to minimize Δ_{Le}. With this method the correspondent vector is split in the following way:

$$V_1=[x(1)\ x(2)\ x(3)\ \ldots\ x(Nt/2)];\ V_2=[x(Nt/2+1)\ x(Nt2+2)\ \ldots\ x(Nt)] \quad (16)$$

The vector V_1 is scrolled along negative x-axis as well as V_2 is scrolled along the positive x-axis. The variables E and F are used to control the limits of the scrolling movement. This technique will give an optimized topology if the uniformity, uniqueness and $\Delta_{Le} \leq \varepsilon$ conditions meet, otherwise the next technique will proceed.

7. - Expansion and Scrolling technique. According to TR, the vector obtained with "fsolve" is improved in order to minimize Δ_{Le}. This method is a mixed method between the two techniques mentioned above, and gives an optimized topology if the uniformity, uniqueness and $\Delta_{Le} \leq \varepsilon$ conditions meet; otherwise the technique is not able to achieve an optimized topology for the given number of different phase centers positions. In that situation a brute force algorithm will start considering all the possible positions according to *xPos* for Nt and Nr elements.

The advantage of this method lies on the fact that if a valid solution is obtained then the condition $\Delta_{Le} \leq \varepsilon$ will meet. Nevertheless, the time these techniques need to obtain the optimum topology is much higher than the time LLS or NLLS (itself) methods need. By using appropriate values of *C*, *D*, *E* and *F*, the execution time will decrease. This can be done by using an adaptive algorithm whose convergence will supply the appropriate values for these variables. However, this task will not be discussed in this document. The symmetry situation is implemented by imposing an even number of transmitters and receivers so that Nt=Nr[1], if this condition is not met, the technique will not reach a feasible solution.

1.4. Genetic Algorithms.

The genetic algorithm is a method for solving optimization problems that is based on natural selection, the process that drives biological evolution. The genetic algorithm repeatedly modifies a population of individual solutions. At each step, the genetic algorithm selects individuals at random from the current population to be parents and uses them to produce the children for the next generation. Over successive generations, the population evolves toward an optimal solution. Usually they are systematic methods to find and optimize a solution of a problem by codifying it in chromosomes (usually strings with 0's and 1's). This method tries to optimize a fitness function F (x1, x2, x3… xn), where (x1, x2… xn) are coded by a chromosome. The typical operations among chromosomes are: evaluation (fitness), selection, mating and mutation. The method of codification can change the efficiency of the genetic algorithm:

- Generally, related parameters should be close in the chromosome.
- There are one-dimensional and two-dimensional chromosomes.
- The chromosomes can have dynamic or static length.

Common terminology in GA's.

Individuals.

An individual is any point to which you can apply the fitness function. The value of the fitness function for an individual is its score. An individual is sometimes referred to as a genome and the vector entries of an individual as genes.

Populations and Generations.

A *population* is an array of individuals. For example, if the size of the population is 100 and the number of variables in the fitness function is 3, you represent the population by a 100-by-3 matrix.

At each iteration, the genetic algorithm performs a series of computations on the current population to produce a new population. Each successive population is called a new generation.

Diversity.

Diversity refers to the average distance between individuals in a population. A population has high diversity if the average distance is large; otherwise it has low diversity. Diversity is essential to the genetic algorithm because it enables the algorithm to search a larger region of the space.

Fitness Values and Best Fitness Values.

The fitness value of an individual is the value of the fitness function for that individual. Because the toolbox finds the minimum of the fitness function, the best fitness value for a population is the smallest fitness value for any individual in the population.

[1] The special topology Nt=2 and Nr (or vice versa) will always find an optimum solution

Parents and Children.

To create the next generation, the genetic algorithm selects certain individuals in the current population, called parents, and uses them to create individuals in the next generation, called children. Typically, the algorithm is more likely to select parents that have better fitness values. The algorithm selects individuals in the current population, called parents, who contribute their genes (the entries of their vectors) to their children. The algorithm most likely selects individuals that have better fitness values as parents. The genetic algorithm creates three types of children for the next generation:

1. - Elite children are the individuals in the current generation with the best fitness values.

2. - Crossover children are created by combining the vectors of a pair of parents. At each coordinate of the child vector, the default crossover function randomly selects an entry, or gene, at the same coordinate from one of the two parents and assigns it to the child.

3. - Mutation children are created by introducing random changes, or mutations, to a single parent. By default, the algorithm adds a random vector from a Gaussian distribution to the parent.

Scheme of a simple GA.

1) Begin.
    a. End=false.
2) Generate an initial population.
3) Compute the function F (x1, x2, x3… xn) to maximize for each individual.
4) WHILE NOT End=true DO
    a. FOR size(population) DO
    b. BEGIN
        i. Mating Phase: Selection of 2 individuals of the previous generation for mating (probability of selection must be proportional to the value of the function F for each individual).
        ii. Mate the two selected individuals with a certain probability to obtain at least two descendants.
        iii. Mutate both descendants with a certain probability.
        iv. Compute the function F for the new individuals.
        v. Insert the individuals in the new generation.
    c. END
    d. IF F reaches a maximum THEN
        i. End=TRUE;
    e. END
5) END

The GA can use the following conditions to determine when to stop:

1. – Generations: The algorithm stops when the number of generations reaches a determined value “Generations”.
2. - Time limit: The algorithm stops after running for an amount of time in seconds equal to “Time limit”.
3. - Fitness limit: The algorithm stops when the value of the fitness function for the best point in the current population is less than or equal to a limit value “Fitness”.
4. - Stall generations: The algorithm stops if there is no improvement in the objective function for a sequence of consecutive generations of length “Stall generations”.
5. - Stall time limit: The algorithm stops if there is no improvement in the objective function during an interval of time in seconds equal to “Stall time limit”.

It is necessary to consider the size of the population, the end conditions of the algorithm and other parameters (relative probabilities, deterministic parameters) when the selection, mating and mutation operations are being executed.

The decoding of the chromosomes and the evaluation of the fitness function F are common operations in the evaluation phase of the algorithm. This phase must be done for all the individuals of the population. After that, the chromosomes are selected following the criterions to optimize/maximize the fitness function.

In the mating phase, the exchange among all the chromosomes of the population is made (crossover) according to a fixed number of ‘sons’.

More complex is the mutation phase, where the value (or one/more bit/s) of the chromosome can be changed. Usually the change of the value is determined with a certain probability. This probability should be small; otherwise the convergence of the algorithm could be never reached.

The antenna topology optimization has been made using different variants of the genetic algorithms. The MIMO array has N_T transmit and N_R receive antennas. An additional constraint is that of avoiding as much as possible pairs of closely spaced Tx/Rx antennas. This is needed in order to reduce the coupling or cross-talk between Tx and Rx channels. The proposed optimization of the topology is made forcing a uniform arrangement of the N_R receivers. The genetic algorithms are targeted to identify the positions of the N_T antennas fulfilling the uniformity and uniqueness conditions for the MIMO $N_T \times N_R$ phase centers. The fitness functions used in the genetic algorithms are:

$$F(rx)=\frac{m_H}{\sigma_H}=\frac{mean(diff(PC))}{std(diff(PC))} \tag{17}$$

$$F(rx)=length(unique(PC))*length(find(H=\mathrm{mode}(H)))\frac{100}{NantT*NantR-1} \tag{18}$$

$$F(rx)=\frac{m_H}{\sigma_H}=\frac{\max(diff(PC))-\min(diff(PC))}{mean(diff(PC))} \tag{19}$$

where m and σ denote, respectively the mean and standard deviation of the vector H, which is computed (in Matlab) as H=diff(PC), with PC being the vector containing the coordinates of the T_x/R_x phase centers sorted in ascending order. The advantage of using these fitness functions is that they have an extremely low computational cost and the synthesis of the radar image is not needed. This gives the possibility to investigate a massive set of possible antenna topologies, typically millions.

Six different genetic algorithms were designed. In general all of them will try to calculate 2, 3 or Nt/2 variables according to the following pairs of transmitters and receivers vectors:

Tx1 = cat (2, linspace (-C, -B, Nt/2), linspace (B, C, Nt/2))
Rx1 = linspace (-A, A, Nr)

Tx2 = [x(1) –x(2) –x(3) … x(Nt/2)] U [x(1) x(2) x(3) … x(Nt/2)]
Rx2 = linspace (-K_1, K_1, Nr)

Tx3 = sort (cat (2, linspace (C-B/2,C+B/2,Nt/2),-linspace (C-B/2,C+B/2,Nt/2)))
Rx3 = linspace (-A/2, A/2, Nr)

The schemes of the vectors are showed in figure 6, the vectors Tx1 and Rx1 are represented by figure 6.a, whereas Tx3 and Rx3 are represented by figure 6.b.

We can divide the six GA's into two different groups considering the internal calculations of the algorithm. Each group will have three different variants according to the number of variables of the vector T_x (2, 3 or Nt/2) to calculate. The first group of the genetic algorithms was designed by using the m-function 'ga'. To use the GA on an unconstrained/constrained problem at the Matlab command line, the user can call the function 'ga' with the syntax:

[x, fval, exitflag, output, population, scores]=ga (@fitnessfun,nvars,A,b,Aeq,beq,LB,UB,nonlcon,options) (20)

where:

- *@fitnessfun* is a handle to the fitness function.
- *nvars* is the number of independent variables for the fitness function.
- *options* is a structure containing options for the genetic algorithm created with the 'gaoptimset' command. If the user does not pass in this argument, 'ga' uses its default options.
- *A* and *b* are a matrix and a vector for inequality constraints.
- *Aeq* and *beq* are a matrix and a vector for equality constraints.
- *LB* and *UB* are the lower and upper bound on x.
- *x* is the point at which the final value is attained.
- *fval* is the final value of the fitness function.
- *exitflag*

  '1': Average cumulative change in value of the fitness function over options.StallGenLimit generations less than options.TolFun and constraint violation less than options.TolCon.

  '2': Fitness limit reached and constraint violation less than options.TolCon.

  '3': The value of the fitness function did not change in options.StallGenLimit generations and constraint violation less than options.TolCon.

  '4': Magnitude of step smaller than machine precision and constraint violation less than options.TolCon.

  '0': Maximum number of generations exceeded.

  '-1': Optimization terminated by the output or plot function.

  '-2': No feasible point found.

  '-4': Stall time limit exceeded.

  '-5': Time limit exceeded.
- *output*

  'randstate': The state of rand, the MATLAB random number generator, just before the algorithm started.

‘generations’: The number of generations.

‘computed.funccount’: The number of evaluations of the fitness functions.

‘message’: The reason the algorithm terminated.

‘maxconstraint’: Maximum constraint violation, if any.

- *population* is a matrix whose rows are the final population.
- *scores* is a vector with the scores of the final population.

Alternatively, the Genetic Algorithm Tool can be invoked by entering the command ‘gatool’.

The second group was designed according to the schemes presented in figure 6. These GA’s consider a better approximation of the solution of the problem by applying a mutation function similar to the schemes in figure 6. Nevertheless the required time to get a ‘valid’ solution can be extremely high compared with that needed in the first GA’s group. The procedure of these GA’s is as follows:

- The R_x coordinates are fixed with a “linspace” m-function to reduce the population of the algorithm as it was showed in the previous schemes.
- The T_x vector will be considered symmetrical respect to y-axis and the T_x coordinates will be forced to have a value in the interval [0.7, 1.2] (for the positive x-axis). To calculate the vector T_x, the user can choose the number of variables to calculate in the optimization process (2, 3 or Nt/2).
- The genetic algorithm will use 16 chromosomes (4 parents and 12 sons) coded with real numbers. The number of elite children is 2.
- The mutation process will be performed every 10 generations. The mutation process will add/take out a random number or constant to the elite children.
- To reduce the individuals the 12 worst chromosomes will be eliminated according to their value of the fitness function; therefore in the mating process 12 new chromosomes will be created.
- The mating process for the Nt/2 case will consider the elite children and one crossover point among the parents. The crossover point will be calculated randomly.
- The three GA’s will finish when the fitness function reaches the maximum value according to equation (10), e.g. (Nt=Nr=16, F= 25600). If the other fitness functions are used, then another terminated condition must be defined. The GA should be changed to minimize instead of maximizing in case the fitness function of the equation 11 is used.

1.5. Direct Search Algorithm (DS).

Direct search is a method for solving optimization problems that does not require any information about the gradient of the objective function. Unlike more traditional optimization methods that use information about the gradient or higher derivatives to search for an optimal point, direct search algorithm searches a set of points around the current point, looking for one where the value of the objective function is lower than the value at the current point. You can use direct search methods to solve a variety of optimization problems that are not well suited for standard optimization algorithms, including problems in which the objective function is discontinuous, non-differentiable, stochastic, or highly non-linear. We will focus on a special class of direct search algorithms called pattern search algorithms. A pattern search algorithm computes a sequence of points that get closer to the optimal point. The pattern search function can solve a linearly constrained minimization problem of the form:

$$\min(f(x)) \quad \forall x \quad / \quad Ax \leq b, \quad A_{eq}x = b_{eq}, \quad LB \leq x \leq UB \qquad \textbf{(21)}$$

- At each step, the algorithm searches a set of points, called a mesh, around the current point, the point computed at the previous step of the algorithm.

- The algorithm forms the mesh by adding the current point to a scalar multiple of a fixed set of vectors called a pattern.

- If the algorithm finds a point in the mesh that improves the objective function at the current point, the new point becomes the current point at the next step of the algorithm.

A pattern is a collection of vectors that the algorithm uses to determine which points to search at each iteration. At each step, the pattern search algorithm searches a set of points, called a mesh, for a point that improves the objective function. The algorithm forms the mesh by:

1. Multiplying the pattern vectors by a scalar, called the *mesh size*.

2. Adding the resulting vectors to the *current point* – the point with the best objective function value found at the previous step.

The pattern vector that produces a mesh point is called its *direction*. At each step, the algorithm polls the points in the current mesh by computing their objective function values. By default the algorithm stops polling the mesh points as soon as it finds a point whose objective function value is less than that of the current point. The poll is then called *successful* and that point becomes the current point at the next iteration. If a complete poll is required, then the algorithm will compute the objective function values at all mesh points. After a *successful poll*, the algorithm multiplies the current mesh size by 2, the default value of the **Mesh Expansion factor**. Because the initial mesh size is 1, at the second iteration the mesh size is 2. If the algorithm fails to find a point that improves the objective function, the poll is called *unsuccessful* and the current point stays the same at the next iteration. After an *unsuccessful poll*, the algorithm multiplies the current mesh size by 0.5, the default value of the **Mesh Contraction factor**. The algorithm then polls with a smaller mesh size.

In addition to polling the mesh points, the pattern search algorithm can perform an optional step at every iteration, called search. At each iteration, the search step applies another optimization method to the current point. If this search does not improve the current point, the poll step is performed.

The algorithm stops when any of the following conditions occurs:

1. The mesh size is less than **Mesh tolerance**.

2. The number of iterations performed by the algorithm reaches the value of **Max iteration**.

3. The total number of objective function evaluations performed by the algorithm reaches the value of **Max function evaluations**.

4. The distance between the point found at one successful poll and the point found at the next successful poll is less than **X tolerance**.

5. The change in the objective function from one successful poll to the next successful poll is less than **Function tolerance**.

6. The **Bind tolerance** option, which is used to identify active constraints for constrained problems, is not used as a stopping criterion.

2. - Results: Optimum Topologies.

Some results of all the methods commented in this document are presented in this section. First the graphic results of the topologies are showed (not for all examples), after that the user can check the characteristics of the topology written in a table, and at the end the numeric values of T_x/R_x are given.

2.1. Special Case Nt=2/Nr=x.

Tx = [-0.8348 0.8348]

Rx = linspace(-0.7652,0.7652,12)

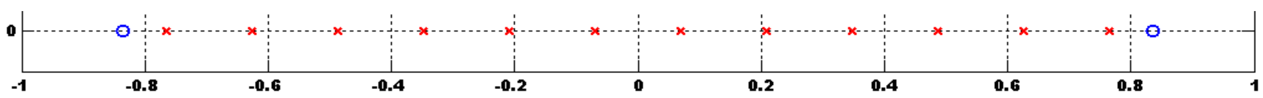


Figure 7.- Optimum Topology Nt=2, Nr=12 (NLLS).

Scheme	A	B	C	Δt	Δr	Unique PC	Unif	ΔPC	Ltx	Lrx	Lant	Lpc	Free Alias Region
2x10	0.8348	-	-	1.6696	0.1391	24	100	0.0695	1.6696	1.5304	1.6696	1.6000	yes

Tx = [-0.8381 0.8381]

Rx = linspace(-0.7619,0.7619,11)

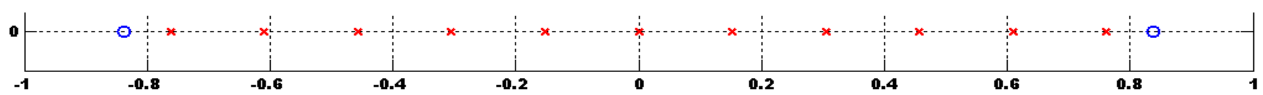


Figure 8.- Optimum Topology Nt=2, Nr=11 (NLLS).

Scheme	A	B	C	Δt	Δr	Unique PC	Unif	ΔPC	Ltx	Lrx	Lant	Lpc	Free Alias Region
2x11	0.8381	-	-	1.6762	0.1524	22	100	0.0761	1.6762	1.5238	1.6762	1.6000	yes

2.2. Nt≠Nr.

Tx = [-1.3984 -1.1843 -0.9701 -0.7559 -0.5417 -0.3276 -0.1134 0.1008 0.3150 0.5291 0.7433 0.9575 1.1717 1.3858 1.6000]

Rx = [-0.2016 -0.1890 -0.1764 -0.1638 -0.1512 -0.1386 -0.1260 -0.1134 -0.1008 -0.0882 -0.0756 -0.0630 -0.0504 -0.0378 -0.0252 -0.0126 0]

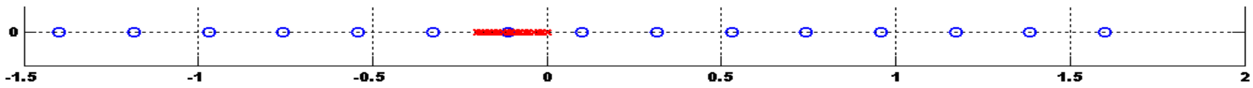


Figure 9.- Better Topology calculated with Nt=15, Nr=17 (no-constrained LLS).

Scheme	A	B	C	Δt	Δr	Unique PC	Unif	ΔPC	Ltx	Lrx	Lant	Lpc	Free Alias Region
15x17	-	-	-	0.2142	0.0126	255	100	0.0062	2.9984	0.2016	2.9984	1.6000	yes

Tx = [-1.1000 -0.8703 -0.6407 -0.4110 -0.1813 0.0483 0.2780 0.5077 0.7373 0.9670 1.1967 1.4263 1.6560 1.8856]

Rx = [-0.5000 -0.4847 -0.4694 -0.4541 -0.4388 -0.4234 -0.4081 -0.3928 -0.3775 -0.3622 -0.3469 -0.3316 -0.3163 -0.3010 -0.2856]

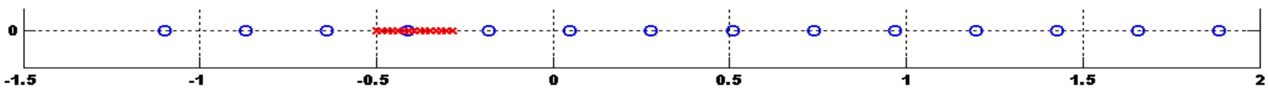


Figure 10.- Better Topology calculated with Nt=14, Nr=15 (constrained LLS).

Scheme	A	B	C	Δt	Δr	Unique PC	Unif	ΔPC	Ltx	Lrx	Lant	Lpc	Free Alias Region
14x15	-	-	-	0.2297	0.0153	210	100	0.0076	2.9856	0.7856	2.9856	1.6000	no

2.3. Nt=Nr.

Tx = linspace(-0.9091,0.9091,10)

Rx = cat(2,linspace(-1.0908,-0.9292,5), linspace(0.9292,1.0908,5))

*diff(Rx) = [0.0404 0.0404 0.0404 0.0404 1.8584 0.0404 0.0404 0.0404 0.0404]

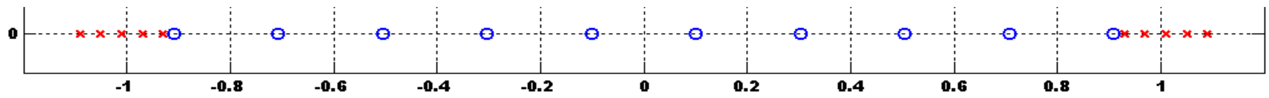


Figure 11.- Optimum Topology Nt=10, Nr=10 (NLLS).

Scheme	A	B	C	Δt	Δr	Unique PC	Unif	ΔPC	Ltx	Lrx	Lant	Lpc	Free Alias Region
10x10	0.9091	1.0908	0.9292	0.202	*	100	100	0.0202	1.8182	0.1816* (1/2)	2.1816	1.9999	yes

Tx = linspace(-0.9231,0.9231,12)

Rx = cat(2,linspace(-1.0768,-0.937,6), linspace(0.937,1.0768,6))

*diff(Rx) = [0.028 0.028 0.028 0.028 0.028 1.874 0.028 0.028 0.028 0.028 0.028]

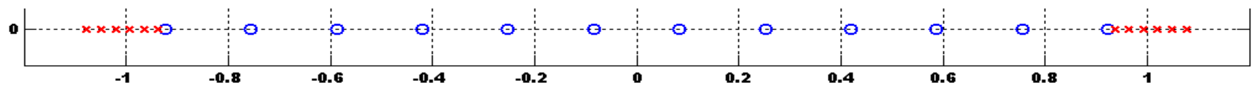


Figure 12.- Optimum Topology Nt=12, Nr=12 (NLLS).

Scheme	A	B	C	Δt	Δr	Unique PC	Unif	ΔPC	Ltx	Lrx	Lant	Lpc	Free Alias Region
12x12	-0.9231	-1.0768	-0.937	0.1678	*	144	100	0.0139	1.8462	0.1398* (1/2)	2.1536	1.9999	no

Tx = linspace(-0.9333, 0.9333,14)

Rx = cat(2,linspace(-1.0663,-0.9433,7),linspace(0.9433,1.0663,7))

*diff(Rx) = [0.0205 0.0205 0.0205 0.0205 0.0205 0.0205 1.8866 0.0205 0.0205 0.0205 0.0205 0.0205 0.0205]

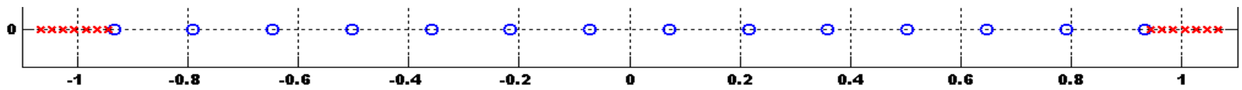


Figure 13.- Optimum Topology Nt=14, Nr=14 (NLLS).

Scheme	A	B	C	Δt	Δr	Unique PC	Unif	ΔPC	Ltx	Lrx	Lant	Lpc	Free Alias Region
14x14	0.9333	1.0663	0.9433	0.1436	*	196	100	0.0102	1.8666	0.12 $*_{(1/2)}$	2.1326	1.9996	no

Tx = linspace(-0.9412,0.9412,16)

Rx = cat(2,linspace(-1.0586,-0.9488,8), linspace(0.9488,1.0586,8))

diff(Rx) = [0.0157 0.0157 0.0157 0.0157 0.0157 0.0157 0.0157 1.8976 0.0157 0.0157 0.0157 0.0157 0.0157 0.0157 0.0157]

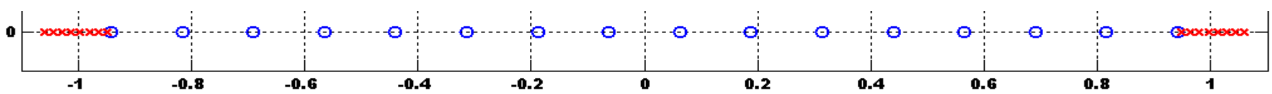


Figure 14.- Optimum Topology Nt=16, Nr=16 (NLLS).

Scheme	A	B	C	Δt	Δr	Unique PC	Unif	ΔPC	Ltx	Lrx	Lant	Lpc	Free Alias Region
16x16	0.9412	1.0586	0.9488	0.1255	*	256	100	0.0078	1.8824	0.1098 $*_{(1/2)}$	2.1172	1.9998	no

Tx = linspace(-0.9474,0.9474,18)

Rx = cat(2,linspace(-1.0523,-0.9532,9), linspace(0.9532,1.0523,9))

*diff(Rx) = [0.0124 0.0124 0.0124 0.0124 0.0124 0.0124 0.0124 0.0124 1.9064 0.0124 0.0124 0.0124 0.0124 0.0124 0.0124 0.0124 0.0124]

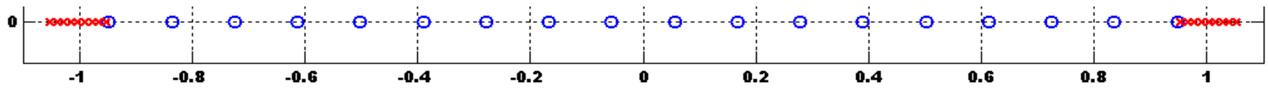


Figure 15.- Optimum Topology Nt=18, Nr=18 (NLLS).

Scheme	A	B	C	Δt	Δr	Unique PC	Unif	ΔPC	Ltx	Lrx	Lant	Lpc	Free Alias Region
18x18	0.9474	1.0523	0.9532	0.1115	*	324	100	0.0061	1.8948	0.0991 *(1/2)	2.1046	1.9997	no

Tx = linspace(-0.9524,0.9524,20)

Rx = cat(2,linspace(-1.0472,-0.957,10), linspace(0.957, 1.0472,10))

*diff(Rx) = [0.01 0.01 0.01 0.01 0.01 0.01 0.01 0.01 0.01 1.914 0.01 0.01 0.01 0.01 0.01 0.01 0.01 0.01 0.01]

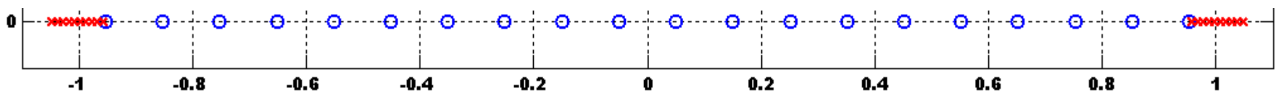


Figure 16.- Optimum Topology Nt=20, Nr=20 (NLLS).

Scheme	A	B	C	Δt	Δr	Unique PC	Unif	ΔPC	Ltx	Lrx	Lant	Lpc	Free Alias Region
20x20	0.9524	1.0472	0.957	0.1003	*	400	100	0.005	1.9048	0.0902 *(1/2)	2.0944	1.9996	no

2.4. Brute Force Algorithm.

With this algorithm all the possible results can be calculated. In first place we will show the results of all possible optimum topologies for the first configuration (vectors T_x/R_x) shown in section 1.A. Each value in the row labeled as 'A' has a correspondent value in 'B' and 'C', for example, for A=1.0901, B = 0.9300 and C=0.9090. After that the results for the second configuration are shown for Nt=Nr=12, 14 and 16.

Nt=Nr=10 Configuration 1	*Optimum Topologies with BF algorithm* *resolution=0.001* *A=0.8:resolution:1.2* *B=0.7:resolution:1* *C=0.7:resolution:1*
A	1.0821 1.0831 1.0841 1.0851 1.0861 1.0871 1.0871 1.0881 1.0881 1.0891 1.0891 1.0901 **1.0901** 1.0911 1.0921 1.0931 1.0931 1.0941 1.0941 1.0951 1.0951 1.0961 1.0971 1.0981 1.0981 1.0991 1.0991
B	0.9220 0.9230 0.9240 0.9250 0.9260 0.9260 0.9270 0.9270 0.9280 0.9280 0.9290 0.9290 **0.9300** 0.9300 0.9310 0.9310 0.9320 0.9320 0.9330 0.9330 0.9340 0.9340 0.9350 0.9350 0.9360 0.9360 0.9370
C	0.9010 0.9020 0.9030 0.9040 0.9050 0.9050 0.9060 0.9060 0.9070 0.9070 0.9080 0.9080 **0.9090** 0.9090 0.9100 0.9100 0.9110 0.9110 0.9120 0.9120 0.9130 0.9130 0.9140 0.9140 0.9150 0.9150 0.9160

Nt=Nr=12 Configuration 1	*Optimum Topologies with BF algorithm* *resolution=0.001* *A=0.8:resolution:1.2* *B=0.7:resolution:1* *C=0.7:resolution:1*
A	1.0501 1.0511 1.0511 1.0521 1.0521 1.0531 1.0541 1.0551 1.0551 1.0561 1.0561 1.0571 1.0571 1.0581 1.0581 1.0591 1.0591 1.0601 1.0611 1.0621 1.0621 1.0631 1.0631 1.0641 1.0641 1.0651 1.0651 1.0661 1.0671 1.0681 1.0691 1.0701 1.0711 1.0721 1.0801 1.0811 1.0821 1.0831 1.0841 1.0851 1.0861 1.0871 1.0871 1.0881 1.0881 1.0891 1.0891 1.0901 1.0901 1.0911
B	0.9140 0.9140 0.9150 0.9150 0.9160 0.9160 0.9170 0.9170 0.9180 0.9180 0.9190 0.9190 0.9200 0.9200 0.9210 0.9210 0.9220 0.9220 0.9230 0.9230 0.9240 0.9240 0.9250 0.9250 0.9260 0.9260 0.9270 0.9270 0.9280 0.9290 0.9300 0.9310 0.9320 0.9330 0.9400 0.9410 0.9420 0.9430 0.9440 0.9450 0.9460 0.9460 0.9470 0.9470 0.9480 0.9480 0.9490 0.9490 0.9500 0.9500
C	0.9000 0.9000 0.9010 0.9010 0.9020 0.9020 0.9030 0.9030 0.9040 0.9040 0.9050 0.9050 0.9060 0.9060 0.9070 0.9070 0.9080 0.9080 0.9090 0.9090 0.9100 0.9100 0.9110 0.9110 0.9120 0.9120 0.9130 0.9130 0.9140 0.9150 0.9160 0.9170 0.9180 0.9190 0.9250 0.9260 0.9270 0.9280 0.9290 0.9300 0.9310 0.9310 0.9320 0.9320 0.9330 0.9330 0.9340 0.9340 0.9350 0.9350

Nt=Nr=14 Configuration 1	*Optimum Topologies with BF algorithm resolution=0.001 A=0.8:resolution:1.2 B=0.7:resolution:1 C=0.7:resolution:1*									
A	1.0291	1.0301	1.0311	1.0321	1.0331	1.0421	1.0431	1.0441	1.0451	1.0461
	1.0471	1.0481	1.0491	1.0501	1.0501	1.0511	1.0511	1.0521	1.0521	1.0531
	1.0531	1.0541	1.0541	1.0551	1.0561	1.0571	1.0571	1.0581	1.0581	1.0591
	1.0591	1.0601	1.0601	1.0611	1.0611	1.0621	1.0621	1.0631	1.0641	1.0651
	1.0651	1.0661	1.0661	1.0671	1.0671	1.0681	1.0681	1.0691	1.0691	1.0701
			1.0711	1.0721	1.0721	1.0731	1.0731	1.0741		
B	0.9100	0.9110	0.9120	0.9130	0.9140	0.9220	0.9230	0.9240	0.9250	0.9260
	0.9270	0.9280	0.9290	0.9290	0.9300	0.9300	0.9310	0.9310	0.9320	0.9320
	0.9330	0.9330	0.9340	0.9340	0.9350	0.9350	0.9360	0.9360	0.9370	0.9370
	0.9380	0.9380	0.9390	0.9390	0.9400	0.9400	0.9410	0.9410	0.9420	0.9420
	0.9430	0.9430	0.9440	0.9440	0.9450	0.9450	0.9460	0.9460	0.9470	0.9470
			0.9480	0.9480	0.9490	0.9490	0.9500	0.9500		
C	0.9000	0.9010	0.9020	0.9030	0.9040	0.9110	0.9120	0.9130	0.9140	0.9150
	0.9160	0.9170	0.9180	0.9180	0.9190	0.9190	0.9200	0.9200	0.9210	0.9210
	0.9220	0.9220	0.9230	0.9230	0.9240	0.9240	0.9250	0.9250	0.9260	0.9260
	0.9270	0.9270	0.9280	0.9280	0.9290	0.9290	0.9300	0.9300	0.9310	0.9310
	0.9320	0.9320	0.9330	0.9330	0.9340	0.9340	0.9350	0.9350	0.9360	0.9360
			0.9370	0.9370	0.9380	0.9380	0.9390	0.9390		

Nt=Nr=16 Configuration 1	*Optimum Topologies with BF algorithm resolution=0.001 A=0.8:resolution:1.2 B=0.7:resolution:1 C=0.7:resolution:1*										
A	1.0121	1.0131	1.0141	1.0151	1.0151	1.0161	1.0161	1.0171	1.0171	1.0181	1.0181
	1.0191	1.0191	1.0201	1.0201	1.0211	1.0211	1.0221	1.0231	1.0241	1.0241	1.0251
	1.0251	1.0261	1.0261	1.0271	1.0271	1.0281	1.0281	1.0291	1.0291	1.0301	1.0311
	1.0321	1.0321	1.0331	1.0331	1.0341	1.0341	1.0351	1.0351	1.0361	1.0361	1.0371
	1.0371	1.0381	1.0381	1.0391	1.0401	1.0411	1.0411	1.0421	1.0421	1.0431	1.0431
	1.0441	1.0441	1.0451	1.0451	1.0461	1.0461	1.0471	1.0481	1.0491	1.0491	1.0501
	1.0501	1.0511	1.0511	1.0521	1.0521	1.0531	1.0531	1.0541	1.0541	1.0551	1.0551
		1.0561	1.0571	1.0581	1.0581	1.0591	1.0591	1.0601	1.0601	1.0611	
B	0.9080	0.9080	0.9090	0.9090	0.9100	0.9100	0.9110	0.9110	0.9120	0.9120	0.9130
	0.9130	0.9140	0.9140	0.9150	0.9150	0.9160	0.9160	0.9170	0.9170	0.9180	0.9180
	0.9190	0.9190	0.9200	0.9200	0.9210	0.9210	0.9220	0.9220	0.9230	0.9230	0.9240
	0.9240	0.9250	0.9250	0.9260	0.9260	0.9270	0.9270	0.9280	0.9280	0.9290	0.9290
	0.9300	0.9300	0.9310	0.9310	0.9320	0.9320	0.9330	0.9330	0.9340	0.9340	0.9350
	0.9350	0.9360	0.9360	0.9370	0.9370	0.9380	0.9380	0.9390	0.9390	0.9400	0.9400
	0.9410	0.9410	0.9420	0.9420	0.9430	0.9430	0.9440	0.9440	0.9450	0.9450	0.9460
		0.9460	0.9470	0.9470	0.9480	0.9480	0.9490	0.9490	0.9500	0.9500	
C	0.9000	0.9000	0.9010	0.9010	0.9020	0.9020	0.9030	0.9030	0.9040	0.9040	0.9050
	0.9050	0.9060	0.9060	0.9070	0.9070	0.9080	0.9080	0.9090	0.9090	0.9100	0.9100
	0.9110	0.9110	0.9120	0.9120	0.9130	0.9130	0.9140	0.9140	0.9150	0.9150	0.9160
	0.9160	0.9170	0.9170	0.9180	0.9180	0.9190	0.9190	0.9200	0.9200	0.9210	0.9210
	0.9220	0.9220	0.9230	0.9230	0.9240	0.9240	0.9250	0.9250	0.9260	0.9260	0.9270
	0.9270	0.9280	0.9280	0.9290	0.9290	0.9300	0.9300	0.9310	0.9310	0.9320	0.9320
	0.9330	0.9330	0.9340	0.9340	0.9350	0.9350	0.9360	0.9360	0.9370	0.9370	0.9380
		0.9380	0.9390	0.9390	0.9400	0.9400	0.9410	0.9410	0.9420	0.9420	

Nt=Nr=20 Configuration 1	*Optimum Topologies with BF algorithm resolution=0.001 A=0.8:resolution:1.2 B=0.7:resolution:1 C=0.7:resolution:1*									
A	0.9901	0.9911	0.9911	0.9921	0.9921	0.9931	0.9931	0.9941	0.9951	0.9961
	0.9961	0.9971	0.9971	0.9981	0.9981	0.9991	0.9991	1.0001	1.0001	1.0011
	1.0011	1.0021	1.0021	1.0031	1.0031	1.0041	1.0051	1.0061	1.0061	1.0071
	1.0071	1.0081	1.0081	1.0091	1.0091	1.0101	1.0101	1.0111	1.0111	1.0121
	1.0121	1.0131	1.0131	1.0141	1.0141	1.0151	1.0161	1.0171	1.0171	1.0181
	1.0181	1.0191	1.0191	1.0201	1.0201	1.0211	1.0211	1.0221	1.0221	1.0231
	1.0231	1.0241	1.0241	1.0251	1.0261	1.0271	1.0281	1.0291	1.0301	1.0311
				1.0321	1.0331	1.0341	1.0351			
B	0.9050	0.9050	0.9060	0.9060	0.9070	0.9070	0.9080	0.9080	0.9090	0.9090
	0.9100	0.9100	0.9110	0.9110	0.9120	0.9120	0.9130	0.9130	0.9140	0.9140
	0.9150	0.9150	0.9160	0.9160	0.9170	0.9170	0.9180	0.9180	0.9190	0.9190
	0.9200	0.9200	0.9210	0.9210	0.9220	0.9220	0.9230	0.9230	0.9240	0.9240
	0.9250	0.9250	0.9260	0.9260	0.9270	0.9270	0.9280	0.9280	0.9290	0.9290
	0.9300	0.9300	0.9310	0.9310	0.9320	0.9320	0.9330	0.9330	0.9340	0.9340
	0.9350	0.9350	0.9360	0.9360	0.9370	0.9380	0.9390	0.9400	0.9410	0.9420
				0.9430	0.9440	0.9450	0.9460			
C	0.9000	0.9000	0.9010	0.9010	0.9020	0.9020	0.9030	0.9030	0.9040	0.9040
	0.9050	0.9050	0.9060	0.9060	0.9070	0.9070	0.9080	0.9080	0.9090	0.9090
	0.9100	0.9100	0.9110	0.9110	0.9120	0.9120	0.9130	0.9130	0.9140	0.9140
	0.9150	0.9150	0.9160	0.9160	0.9170	0.9170	0.9180	0.9180	0.9190	0.9190
	0.9200	0.9200	0.9210	0.9210	0.9220	0.9220	0.9230	0.9230	0.9240	0.9240
	0.9250	0.9250	0.9260	0.9260	0.9270	0.9270	0.9280	0.9280	0.9290	0.9290
	0.9300	0.9300	0.9310	0.9310	0.9320	0.9330	0.9340	0.9350	0.9360	0.9370
				0.9380	0.9390	0.9400	0.9410			

Nt=Nr=22 Configuration 1	*Optimum Topologies with BF algorithm resolution=0.001 A=0.8:resolution:1.2 B=0.7:resolution:1 C=0.7:resolution:1*								
A	0.9821	0.9831	0.9831	0.9841	0.9841	0.9851	0.9851	0.9861	0.9861
	0.9871	0.9881	0.9891	0.9901	0.9911	0.9921	0.9931	0.9941	0.9951
	0.9961	0.9971	1.0111	1.0121	1.0131	1.0141	1.0151	1.0161	1.0171
	1.0181	1.0191	1.0201	1.0211	1.0221	1.0221	1.0231	1.0231	1.0241
	1.0241	1.0251	1.0251	1.0261	1.0261	1.0271	1.0271	1.0281	1.0281
			1.0291	1.0291	1.0301	1.0301	1.0311		
B	0.9040	0.9040	0.9050	0.9050	0.9060	0.9060	0.9070	0.9070	0.9080
	0.9080	0.9090	0.9100	0.9110	0.9120	0.9130	0.9140	0.9150	0.9160
	0.9170	0.9180	0.9310	0.9320	0.9330	0.9340	0.9350	0.9360	0.9370
	0.9380	0.9390	0.9400	0.9410	0.9410	0.9420	0.9420	0.9430	0.9430
	0.9440	0.9440	0.9450	0.9450	0.9460	0.9460	0.9470	0.9470	0.9480
			0.9480	0.9490	0.9490	0.9500	0.9500		
C	0.9000	0.9000	0.9010	0.9010	0.9020	0.9020	0.9030	0.9030	0.9040
	0.9040	0.9050	0.9060	0.9070	0.9080	0.9090	0.9100	0.9110	0.9120
	0.9130	0.9140	0.9260	0.9270	0.9280	0.9290	0.9300	0.9310	0.9320
	0.9330	0.9340	0.9350	0.9360	0.9360	0.9370	0.9370	0.9380	0.9380
	0.9390	0.9390	0.9400	0.9400	0.9410	0.9410	0.9420	0.9420	0.9430
			0.9430	0.9440	0.9440	0.9450	0.9450		

Nt=Nr=32 Configuration 1	*Optimum Topologies with BF algorithm* *resolution=0.001* *A=0.8:resolution:1.2* *B=0.7:resolution:1* *C=0.7:resolution:1*									
A	0.9561	0.9571	0.9581	0.9591	0.9591	0.9601	0.9601	0.9611	0.9611	0.9621
	0.9621	0.9631	0.9631	0.9641	0.9641	0.9651	0.9651	0.9661	0.9661	0.9671
	0.9671	0.9681	0.9681	0.9691	0.9691	0.9701	0.9701	0.9711	0.9711	0.9721
	0.9721	0.9731	0.9731	0.9741	0.9751	0.9761	0.9761	0.9771	0.9771	0.9781
	0.9781	0.9791	0.9791	0.9801	0.9801	0.9811	0.9811	0.9821	0.9821	0.9831
	0.9831	0.9841	0.9841	0.9851	0.9851	0.9861	0.9861	0.9871	0.9871	0.9881
	0.9881	0.9891	0.9891	0.9901	0.9911	0.9921	0.9921	0.9931	0.9931	0.9941
	0.9941	0.9951	0.9951	0.9961	0.9961	0.9971	0.9971	0.9981	0.9981	0.9991
	0.9991	1.0001	1.0001	1.0011	1.0011	1.0021	1.0021	1.0031	1.0031	1.0041
		1.0041	1.0051	1.0051	1.0061	1.0061	1.0071	1.0081	1.0091	
B	0.9020	0.9020	0.9030	0.9030	0.9040	0.9040	0.9050	0.9050	0.9060	0.9060
	0.9070	0.9070	0.9080	0.9080	0.9090	0.9090	0.9100	0.9100	0.9110	0.9110
	0.9120	0.9120	0.9130	0.9130	0.9140	0.9140	0.9150	0.9150	0.9160	0.9160
	0.9170	0.9170	0.9180	0.9180	0.9190	0.9190	0.9200	0.9200	0.9210	0.9210
	0.9220	0.9220	0.9230	0.9230	0.9240	0.9240	0.9250	0.9250	0.9260	0.9260
	0.9270	0.9270	0.9280	0.9280	0.9290	0.9290	0.9300	0.9300	0.9310	0.9310
	0.9320	0.9320	0.9330	0.9330	0.9340	0.9340	0.9350	0.9350	0.9360	0.9360
	0.9370	0.9370	0.9380	0.9380	0.9390	0.9390	0.9400	0.9400	0.9410	0.9410
	0.9420	0.9420	0.9430	0.9430	0.9440	0.9440	0.9450	0.9450	0.9460	0.9460
		0.9470	0.9470	0.9480	0.9480	0.9490	0.9490	0.9500	0.9500	
C	0.9000	0.9000	0.9010	0.9010	0.9020	0.9020	0.9030	0.9030	0.9040	0.9040
	0.9050	0.9050	0.9060	0.9060	0.9070	0.9070	0.9080	0.9080	0.9090	0.9090
	0.9100	0.9100	0.9110	0.9110	0.9120	0.9120	0.9130	0.9130	0.9140	0.9140
	0.9150	0.9150	0.9160	0.9160	0.9170	0.9170	0.9180	0.9180	0.9190	0.9190
	0.9200	0.9200	0.9210	0.9210	0.9220	0.9220	0.9230	0.9230	0.9240	0.9240
	0.9250	0.9250	0.9260	0.9260	0.9270	0.9270	0.9280	0.9280	0.9290	0.9290
	0.9300	0.9300	0.9310	0.9310	0.9320	0.9320	0.9330	0.9330	0.9340	0.9340
	0.9350	0.9350	0.9360	0.9360	0.9370	0.9370	0.9380	0.9380	0.9390	0.9390
	0.9400	0.9400	0.9410	0.9410	0.9420	0.9420	0.9430	0.9430	0.9440	0.9440
		0.9450	0.9450	0.9460	0.9460	0.9470	0.9470	0.9480	0.9480	

Nt=Nr=22 Configuration 2	*Optimum Topologies with BF algorithm* *resolution=0.0001* *A=2* *B=0.05:resolution:0.18* *C=0.99:resolution:1.1*				
B	0.0854	0.0861	0.0865	0.0865	0.0866
C	1.0000	1.0000	1.0346	1.0476	1.0476

Nt=Nr=14 Configuration 2	*Optimum Topologies with BF algorithm* *resolution=0.0001* *A=2* *B=0.06:resolution:0.18* *C=0.99:resolution:1.1*				
B	0.1320	0.1320	0.1320	0.1318	0.1318
C	1.0000	1.0220	1.0440	1.0659	1.0769

Nt=Nr=12 Configuration 2	*Optimum Topologies with BF algorithm* *resolution=0.0001* *A=2* *B=0.05:resolution:0.18* *C=1.07:resolution:1.1*
B	0.1507 0.1508 0.1509 0.1510 0.1513 0.1514 0.1515 0.1501 0.1502 0.1503 0.1504 0.1505 0.1506 0.1507 0.1508 0.1509 0.1510 0.1511 0.1512 0.1513 0.1514 0.1515 0.1501 0.1502 0.1503 0.1504 0.1505 0.1506 0.1507 0.1508 0.1509 0.1510 0.1511 0.1512 0.1513 0.1514 0.1515 0.1502 0.1503 0.1504 0.1505 0.1506 0.1507 0.1508 0.1509 0.1510 0.1511 0.1512 0.1513 0.1514 0.1515 0.1504 0.1505 0.1506 0.1507 0.1508 0.1509 0.1510 0.1511 0.1512 0.1513 0.1514 0.1515 0.1506 0.1507 0.1508 0.1509 0.1510 0.1511 0.1512 0.1513 0.1514 0.1515 0.1507 0.1508 0.1509 0.1510 0.1511 0.1512 0.1513 0.1514 0.1515 0.1509 0.1510 0.1511 0.1512 0.1513 0.1514 0.1515 0.1511 0.1512 0.1513 0.1514 0.1515 0.1512 0.1513 0.1514 0.1515 0.1514 0.1515
C	1.0899 1.0899 1.0899 1.0899 1.0899 1.0899 1.0899 1.0900 1.0900 1.0900 1.0900 1.0900 1.0900 1.0900 1.0900 1.0900 1.0900 1.0900 1.0900 1.0900 1.0900 1.0900 1.0901 1.0901 1.0901 1.0901 1.0901 1.0901 1.0901 1.0901 1.0901 1.0901 1.0901 1.0901 1.0901 1.0901 1.0901 1.0902 1.0902 1.0902 1.0902 1.0902 1.0902 1.0902 1.0902 1.0902 1.0902 1.0902 1.0902 1.0902 1.0902 1.0903 1.0903 1.0903 1.0903 1.0903 1.0903 1.0903 1.0903 1.0903 1.0903 1.0903 1.0903 1.0904 1.0904 1.0904 1.0904 1.0904 1.0904 1.0904 1.0904 1.0904 1.0904 1.0905 1.0905 1.0905 1.0905 1.0905 1.0905 1.0905 1.0905 1.0905 1.0906 1.0906 1.0906 1.0906 1.0906 1.0906 1.0906 1.0907 1.0907 1.0907 1.0907 1.0907 1.0908 1.0908 1.0908 1.0908 1.0909 1.0909

Nt=Nr=16 Configuration 2	*Optimum Topologies with BF algorithm* *resolution=0.0001* *A=2* *B=0.07:resolution:0.18* *C=0.99:resolution:1.1*
B	0.1164 0.1166 0.1167
C	1.0000 1.0583 1.0667

Scheme	A	B	C	Δt	Δr	Unique PC	Unif	ΔPC	Ltx	Lrx	Lant	Lpc	Free Alias Region
12x12	0.9010	1.0511	0.915	0.0272*	0.1638	144	100	0.0136	0.1361	1.802	2.1022	1.9521	no

Tx = [-1.0291 -1.0093 -0.9894 -0.9695 -0.9497 -0.9298 -0.9100 0.9100 0.9298 0.9497 0.9695 0.9894 1.0093 1.0291]

Rx = [-0.9000 -0.7615 -0.6231 -0.4846 -0.3462 -0.2077 -0.0692 0.0692 0.2077 0.3462 0.4846 0.6231 0.7615 0.9000]

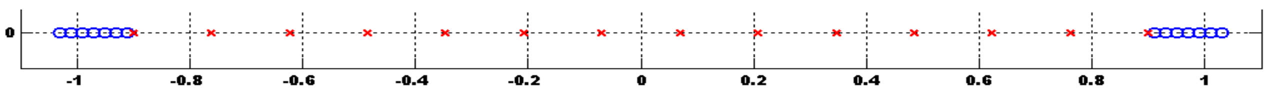


Figure 17.- Optimum Topology Nt=14, Nr=14 (BF).

Scheme	A	B	C	Δt	Δr	Unique PC	Unif	ΔPC	Ltx	Lrx	Lant	Lpc	Free Alias Region
14x14	0.9	1.0291	0.91	0.0199*	0.1385	196	100	0.0099	0.1191	1.8	2.0582	1.9291	no

Tx = [-1.0121 -0.9972 -0.9824 -0.9675 -0.9526 -0.9377 -0.9229 -0.9080 0.9080 0.9229 0.9377 0.9526 0.9675 0.9824 0.9972 1.0121]

Rx = [-0.9000 -0.7800 -0.6600 -0.5400 -0.4200 -0.3000 -0.1800 -0.0600 0.0600 0.1800 0.3000 0.4200 0.5400 0.6600 0.7800 0.9000]

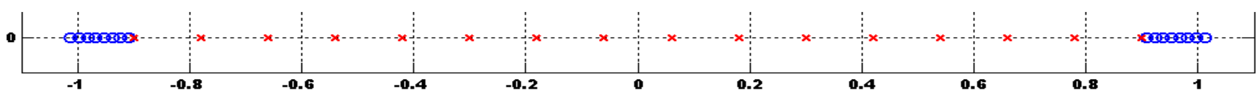


Figure 18.- Optimum Topology Nt=16, Nr=16 (BF).

Scheme	A	B	C	Δt	Δr	Unique PC	Unif	ΔPC	Ltx	Lrx	Lant	Lpc	Free Alias Region
16x16	0.9	1.0121	0.908	0.0149*	0.12	256	100	0.0074	0.1041	1.8	2.0242	1.9121	no

2.5. Genetic Algorithm (Nt=Nr).

OptmTopologyGA3_1D (NantT,NantR)

Tx = [-1.0496 -1.0224 -0.9951 -0.9679 -0.9406 -0.9134 0.9134 0.9406 0.9679 0.9951 1.0224 1.0496]

Rx = [-0.9000 -0.7364 -0.5727 -0.4091 -0.2455 -0.0818 0.0818 0.2455 0.4091 0.5727 0.7364 0.9000]

Number of generations 2101
Elapsed time is 39.495867 seconds.

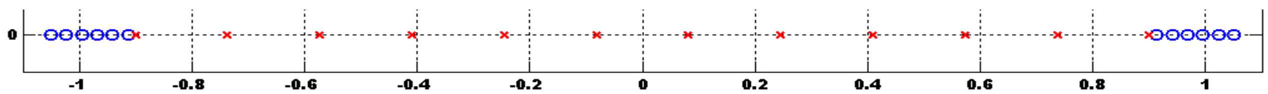


Figure 19.- Optimum Topology Nt=12, Nr=12 (GA V1.3).

Scheme	A	B	C	Δt	Δr	Unique PC	Unif	ΔPC	Ltx	Lrx	Lant	Lpc	Free Alias Region
12x12	0.9	1.0496	0.9134	0.0272*	0.1636*	144	100	0.0136	0.1362	1.8	2.0992	1.9496	yes

Tx = [-1.0284 -1.0086 -0.9887 -0.9689 -0.9491 -0.9292 -0.9094 0.9094 0.9292 0.9491 0.9689 0.9887 1.0086 1.0284]

Rx = [-0.9000 -0.7615 -0.6231 -0.4846 -0.3462 -0.2077 -0.0692 0.0692 0.2077 0.3462 0.4846 0.6231 0.7615 0.9000]

Number of generations 518
Elapsed time is 10.232068 seconds.

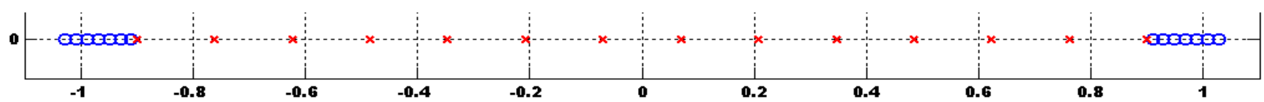


Figure 20.- Optimum Topology Nt=14, Nr=14 (GA V1.3).

Scheme	A	B	C	Δt	Δr	Unique PC	Unif	ΔPC	Ltx	Lrx	Lant	Lpc	Free Alias Region
14x14	0.9	1.0284	0.9094	0.0198*	0.1385*	196	100	0.0099	0.1199	1.8	2.0568	1.9284	no

Tx = [-1.0122 -0.9972 -0.9823 -0.9673 -0.9524 -0.9374 -0.9225 -0.9075 0.9075 0.9225 0.9374 0.9524 0.9673 0.9823 0.9972 1.0122]

Rx = [-0.9000 -0.7800 -0.6600 -0.5400 -0.4200 -0.3000 -0.1800 -0.0600 0.0600 0.1800 0.3000 0.4200 0.5400 0.6600 0.7800 0.9000]

Number of generations 3362

Elapsed time is 65.806075 seconds.

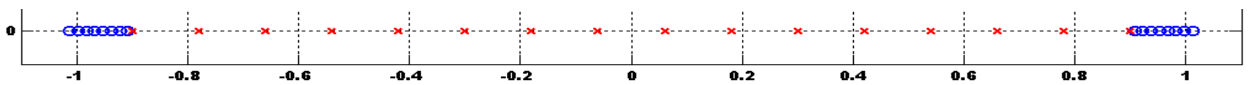


Figure 21.- Optimum Topology Nt=16, Nr=16 (GA V1.3).

Scheme	A	B	C	Δt	Δr	Unique PC	Unif	ΔPC	Ltx	Lrx	Lant	Lpc	Free Alias Region
16x16	0.9	1.0122	0.9075	0.0149*	0.12	256	100	0.0075	0.1047	1.8	2.0244	1.9122	no

OptmTopologyGA2_1D (NantT,NantR)

Tx = [-1.0528 -1.0253 -0.9977 -0.9702 -0.9427 -0.9151 0.9151 0.9427 0.9702 0.9977 1.0253 1.0528]

Rx = [-0.9021 -0.7381 -0.5740 -0.4100 -0.2460 -0.0820 0.0820 0.2460 0.4100 0.5740 0.7381 0.9021]

Number of generations 7922

Elapsed time is 156.768046 seconds.

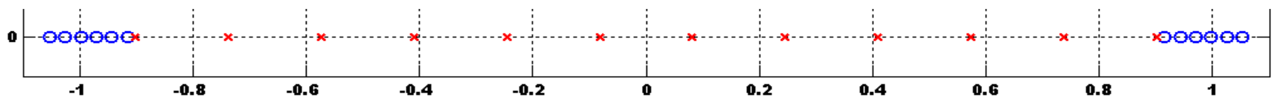


Figure 22.- Optimum Topology Nt=12, Nr=12 (GA V1.2).

Scheme	A	B	C	Δt	Δr	Unique PC	Unif	ΔPC	Ltx	Lrx	Lant	Lpc	Free Alias Region
12x12	0.9021	1.0528	0.9151	0.0275*	0.164*	144	100	0.0137	0.1377	1.8042	2.1056	1.9549	no

Tx = [-1.0677 -1.0473 -1.0269 -1.0065 -0.9861 -0.9657 -0.9453 0.9453 0.9657 0.9861 1.0065 1.0269 1.0473 1.0677]

Rx = [-0.9352 -0.7913 -0.6475 -0.5036 -0.3597 -0.2158 -0.0719 0.0719 0.2158 0.3597 0.5036 0.6475 0.7913 0.9352]

Number of generations 6141

Elapsed time is 121.093416 seconds.

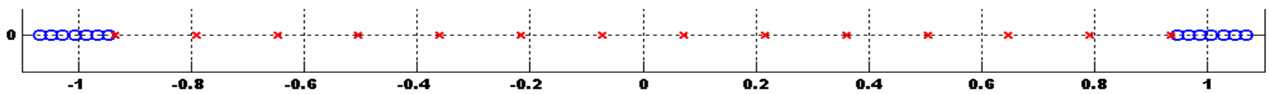


Figure 23.- Optimum Topology Nt=14, Nr=14 (GA V1.2).

Scheme	A	B	C	Δt	Δr	Unique PC	Unif	ΔPC	Ltx	Lrx	Lant	Lpc	Free Alias Region
14x14	0.9352	1.0677	0.9453	0.0204*	0.1439*	196	100	0.0102	0.1224	1.8704	2.1354	2.0029	no

Tx = [-1.0121 -0.9971 -0.9821 -0.9672 -0.9522 -0.9372 -0.9222 -0.9073 0.9073 0.9222 0.9372 0.9522 0.9672 0.9821 0.9971 1.0121]

Rx = [-0.9001 -0.7801 -0.6601 -0.5400 -0.4200 -0.3000 -0.1800 -0.0600 0.0600 0.1800 0.3000 0.4200 0.5400 0.6601 0.7801 0.9001]

Number of generations 18832

Elapsed time is 426.937150 seconds.

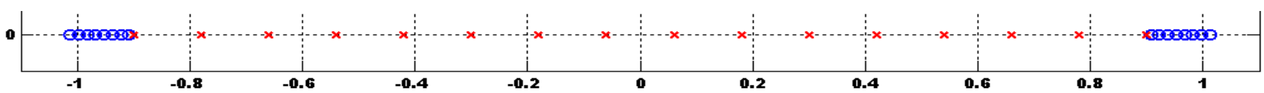


Figure 24.- Optimum Topology Nt=16, Nr=16 (GA V1.2).

Scheme	A	B	C	Δt	Δr	Unique PC	Unif	ΔPC	Ltx	Lrx	Lant	Lpc	Free Alias Region
16x16	0.9001	1.0121	0.9073	0.015*	0.12*	256	100	0.0075	0.1048	1.8002	2.0242	1.9122	no

Other results:

Tx = [-0.9633 -0.9790 -0.9946 -1.0103 -1.0257 -1.0422 -1.0579 -1.0736 0.9633 0.9790 0.9946 1.0103 1.0257 1.0422 1.0579 1.0736]

Rx = linspace(-0.955,0.955,16)

Scheme	A	B	C	Δt	Δr	Unique PC	Unif	ΔPC	Ltx	Lrx	Lant	Lpc	Free Alias Region
16x16	0.955	1.0736	0.9633	0.0157*	0.1273*	256	100	0.0078	0.1103	1.91	2.1472	1.9735	no

Tx = [-1.0738 -1.0516 -1.0294 -1.0072 -0.9850 -0.9628 -0.9406 0.9406 0.9628 0.9850 1.0072 1.0294 1.0516 1.0738]

Rx = [-0.9289 -0.7741 -0.6193 -0.4645 -0.3096 -0.1548 0.0000 0.1548 0.3096 0.4645 0.6193 0.7741 0.9289]

Scheme	A	B	C	Δt	Δr	Unique PC	Unif	ΔPC	Ltx	Lrx	Lant	Lpc	Free Alias Region
14x13	0.955	1.0736	0.9633	0.0222*	0.1548*	182	100	0.0111	0.1332	1.8578	2.1476	0.1449	no

Tx = [-1.0940 -1.0652 -1.0363 -1.0075 -0.9787 -0.9498 -0.9210 0.9210 0.9498 0.9787 1.0075 1.0363 1.0652 1.0940]

Rx = [-0.9065 -0.7050 -0.5036 -0.3022 -0.1007 0.1007 0.3022 0.5036 0.7050 0.9065]

Number of generations 7541
Elapsed time is 148.937491 seconds.

Scheme	A	B	C	Δt	Δr	Unique PC	Unif	ΔPC	Ltx	Lrx	Lant	Lpc	Free Alias Region
14x10	0.955	1.0736	0.9633	0.0288*	0.2015*	140	100	0.0144	0.173	1.813	2.188	2.0005	no

Tx = [-1.0651 -1.0373 -1.0094 -0.9816 -0.9538 -0.9259 0.9259 0.9538 0.9816 1.0094 1.0373 1.0651]

Rx = [-0.9129 -0.7469 -0.5809 -0.4150 -0.2490 -0.0830 0.0830 0.2490 0.4150 0.5809 0.7469 0.9129]

Number of generations 912
Elapsed time is 17.261997 seconds.

Scheme	A	B	C	Δt	Δr	Unique PC	Unif	ΔPC	Ltx	Lrx	Lant	Lpc	Free Alias Region
12x12	0.9129	1.0651	0.9259	0.0278*	0.166*	144	69.9	0.0139*	0.1392	1.8258	2.1302	0.1522	no

Tx = [-1.0450 -1.0250 -1.0049 -0.9848 -0.9648 -0.9447 -0.9247 0.9247 0.9447 0.9648 0.9848 1.0049 1.0250 1.0450]

Rx = [-0.9146 -0.7739 -0.6332 -0.4925 -0.3518 -0.2111 -0.0704 0.0704 0.2111 0.3518 0.4925 0.6332 0.7739 0.9146]

Number of generations 9382
Elapsed time is 196.848746 seconds.

Scheme	A	B	C	Δt	Δr	Unique PC	Unif	ΔPC	Ltx	Lrx	Lant	Lpc	Free Alias Region
14x14	0.9146	1.045	0.9247	0.02*	0.1407*	196	100	0.01	0.1203	1.8292	2.09	1.9596	no

OptmTopologyGA1_1D (NantT,NantR)

Tx = [-1.0564 -1.0408 -1.0252 -1.0096 -0.9940 -0.9784 -0.9627 -0.9471 0.9471 0.9627 0.9784 0.9940 1.0096 1.0252 1.0408 1.0564]

Rx = linspace(-0.9394,0.9394,16)

Scheme	A	B	C	Δt	Δr	Unique PC	Unif	ΔPC	Ltx	Lrx	Lant	Lpc	Free Alias Region
16x16	0.9394	1.0564	0.9471	0.0156*	0.1253	256	100	0.0078	0.1093	1.8788	2.1128	1.9958	no

Tx = [-1.0563 -1.0407 -1.0251 -1.0095 -0.9939 -0.9783 -0.9627 -0.9471 0.9471 0.9627 0.9783 0.9939 1.0095 1.0251 1.0407 1.0563]

Rx = [-0.9387 -0.8135 -0.6884 -0.5632 -0.4381 -0.3129 -0.1877 -0.0626 0.0626 0.1877 0.3129 0.4381 0.5632 0.6884 0.8135 0.9387]

Scheme	A	B	C	Δt	Δr	Unique PC	Unif	ΔPC	Ltx	Lrx	Lant	Lpc	Free Alias Region
16x16	0.9387	1.0563	0.9471	0.0156*	0.1252*	256	100	0.0078	0.1092	1.8774	2.1126	1.995	no

Tx = [-1.0564 -1.0408 -1.0252 -1.0095 -0.9939 -0.9783 -0.9627 -0.9470 0.9470 0.9627 0.9783 0.9939 1.0095 1.0252 1.0408 1.0564]

Rx = [-0.9389 -0.8137 -0.6885 -0.5633 -0.4382 -0.3130 -0.1878 -0.0626 0.0626 0.1878 0.3130 0.4382 0.5633 0.6885 0.8137 0.9389]

Scheme	A	B	C	Δt	Δr	Unique PC	Unif	ΔPC	Ltx	Lrx	Lant	Lpc	Free Alias Region
16x16	0.9389	1.0564	0.947	0.0156*	0.1252	256	100	0.0078	0.1094	1.8778	2.1128	1.9953	no

Tx = [-1.0562 -1.0406 -1.0250 -1.0095 -0.9939 -0.9783 -0.9627 -0.9471 0.9471 0.9627 0.9783 0.9939 1.0095 1.0250 1.0406 1.0562]

Rx = [-0.9388 -0.8136 -0.6885 -0.5633 -0.4381 -0.3129 -0.1878 -0.0626 0.0626 0.1878 0.3129 0.4381 0.5633 0.6885 0.8136 0.9388]

Scheme	A	B	C	Δt	Δr	Unique PC	Unif	ΔPC	Ltx	Lrx	Lant	Lpc	Free Alias Region
16x16	0.9388	1.0562	0.9471	0.0156*	0.1252*	256	100	0.0078	0.1091	1.8776	2.1124	1.995	no

Tx = [-1.0772 -1.0490 -1.0209 -0.9927 -0.9646 -0.9364 0.9364 0.9646 0.9927 1.0209 1.0490 1.0772]

Rx = [-0.9364 -0.7661 -0.5959 -0.4256 -0.2554 -0.0851 0.0851 0.2554 0.4256 0.5959 0.7661 0.9364]

Scheme	A	B	C	Δt	Δr	Unique PC	Unif	ΔPC	Ltx	Lrx	Lant	Lpc	Free Alias Region
12x12	0.9364	1.0772	0.9364	0.0281*	0.1703	144	100	0.0141	0.1408	1.8728	2.1544	2.0136	no

3. Imaging Algorithm Theory.

First, let us describe the scenario to be illuminated by the radar. The targets to be imaged will be three dimensional distributed in the positive Y axis and placed in the scenario defined by the user. The antenna array will lay in the X-Z plane, centered at the origin and pointing to the positive Y direction. In this way, this coordinate system translates the Y axis into the radar ground range direction, the X axis into the radar cross range direction, and Z axis into the vertical direction. The scenario exposed in Figure 26 contains five scatterers (targets) distributed in the positions showed, all of them with a power density of $P_0 = 0$ W/m^{2}and placed in the plane z=0. In this example the imaging plane is defined for x = [-21, 21] and y = [0, 50]. This entails an area of 42×50 meters covered by the system (901 points have been used for imaging in the *x* direction and 801 in the *y* direction).

Afterwards the frequency domain raw data for the scenario and MIMO array of interest is obtained. The signal returned from the scene is obtained by integrating coherently the backscatter of all combinations of transmitters and receivers:

$$raw\,(i,j,f) = \sum_{s=1}^{S} P_0 e^{-j\frac{2\pi(d_1(i,s)+d_2(j,s))}{c}f} \tag{22}$$

where *i* and *j* are the index of the transmitter and receiver respectively. *S* denotes the number of scatterers in the scene, and P_0 is the radar reflectivity of the s*th* scatterer. The variable $d_1(i,s)$ represents the distance between the i*th* transmitter and the s*th* scatterer so subsequently, $d_2(j,s)$ is the distance between the s*th* scatterer and the j*th* receiver of the MIMO array. The imaging technique used the time-domain back propagation algorithm (TDBA). The back propagation algorithm has been chosen because it does not assume the far field approximations and it can be directly applied either to frequency modulated (FMCW) or stepped frequency (SFCW) radars. A FFT or chirpZ transform is then applied to change from the frequency domain to time domain. A Hamming window has been applied prior to the frequency to time transform in the TDBA algorithm in order to reduce the levels of the side lobes of the images obtained with the MIMO array. Finally the radar reflectivity at an arbitrary point *P* in the scene is calculated as follows:

$$I(P) = \sum_{i=1}^{N_T}\sum_{j=1}^{N_R} raw_t(i, j, \tfrac{d_1(i,P)+d_2(j,P)}{c}) \tag{23}$$

where N_T and N_R are, respectively, the number of transmitters and receivers of the array. raw_t is the result of the frequency to time domain transform of the raw data. $d_1(i, (x_p,y_p))$ and $d_2(j, (x_p,y_p))$ are, respectively, the distances between the point (x_p,y_p) and the i*th* transmitter and jth receiver antenna elements of the MIMO array. The formation of the image is made in cylindrical coordinates.

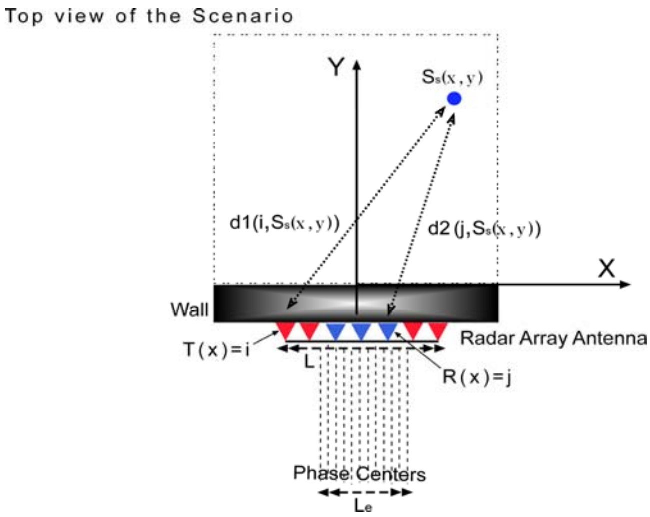
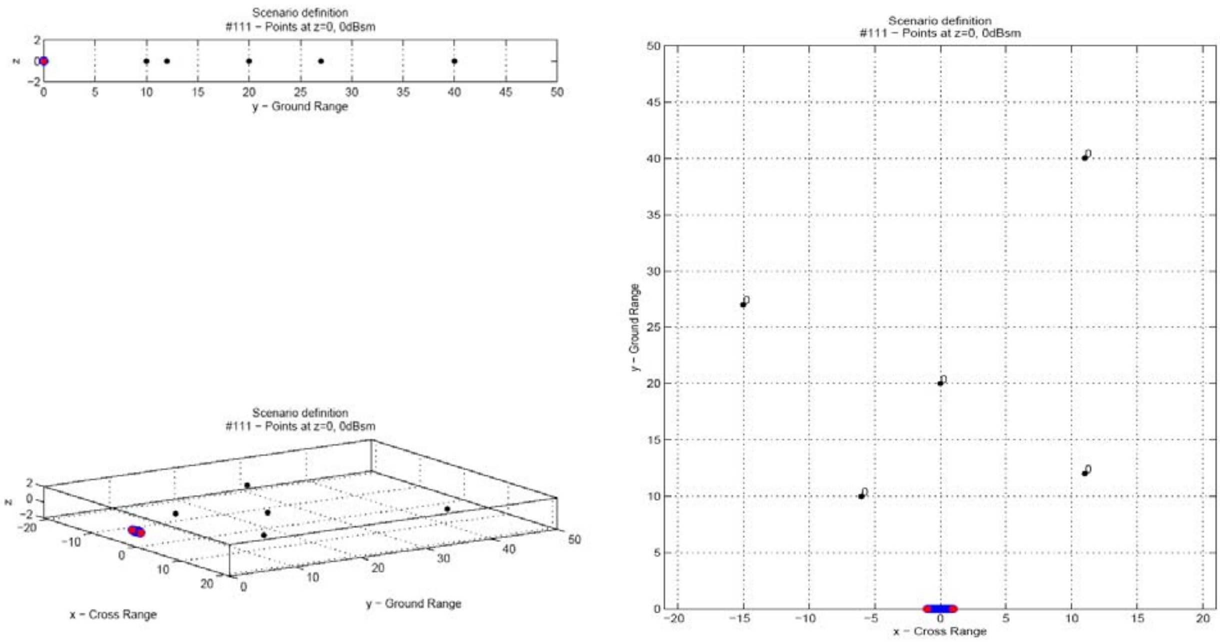


Figure 25.- Image Geometry.



26.a. Lateral and Perspective view. **26.b. Top view.**

Figure 26.- Scenario.

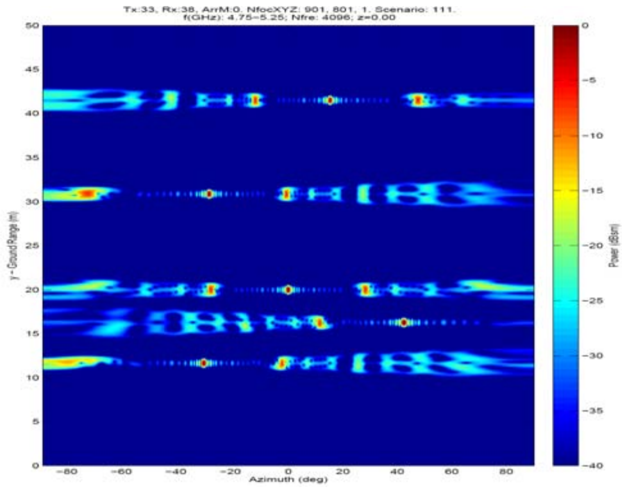
4. Results: MIMO and SAR Images with Optimum Topologies.

Once the performed algorithm identifies the positions of the N_t and N_r antennas fulfilling uniformity and uniqueness condition of the MIMO $N_t \times N_r$ phase centers, the imaging simulator [1] will calculate the SAR and MIMO images. The scenario used through all the simulations for the evaluation of the image can be seen in figures 26. In some simulations a Blackman Harris windowing function along the x-axis of the radar aperture is applied. The windowing function reduces the levels of the side lobes of the antenna in the images obtained with the MIMO array. This window is built considering the sorted phase centers positions (PC) of the MIMO array antenna and using the equation:

$$w(n) = 0.35875 + 0.48829\cos\left(\frac{2\pi}{N}n\right) + 0.14128\cos\left(\frac{4\pi}{N}n\right) - 0.01168\cos\left(\frac{6\pi}{N}n\right) \quad (24)$$

where $-\frac{N}{2} \leq n \leq \frac{N}{2}$, $L=N+1$ is the window length and $\frac{n}{N} = \frac{PC - \min(PC)}{\max(PC) - \min(PC)}$.

Most of the methods can calculate optimum topologies, so the differences among the MIMO/SAR images obtained with these different methods are almost inappreciable. In this way only some results will be shown in the next sub-section for some calculated topologies.



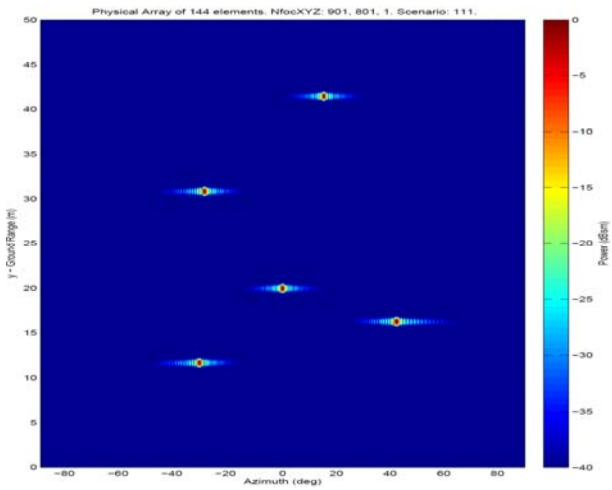


Figure 27.- a. Radar image of a non optimum MIMO 16x16 array.
b. Radar image of the SAR array with 256 elements (Reference image for a 16x16 array).

4.1. Operations with Optimum Topologies.

Once an optimum topology is calculated (T_x/R_x), it should be important to check the feasibility of producing new optimum topologies by applying mathematical operations to the previous one. In this sub-section we are going to demonstrate the needed to execute optimization techniques every time we want to calculate a new topology with different values of Nt and/or Nr starting from a given optimum topology.

We developed a m-function which will perform a set of calculations to carry out this task. In a first attempt, we will apply some calculations to the given topology and allow the m-function to change the length of the original optimum array antenna. After that, we will reproduce the same operations without changing the length of the antenna given at the beginning of the procedure. In this way, the performed operations will be done to the pair T_x/R_x. In general new elements will be added at the right side of the vectors T_x/R_x. This procedure is performed as follows:

- First the array *deltavec=diff(sort(vector))* is obtained (*vector* can represent T_x or R_x).
- The mode of *deltavec* is calculated and found in the array *deltavec.* The new element will be add at the right part of *vector* and separated *mode(deltavec)* from the last element.

The optimum T_x / R_x vectors can be modified by adding transmitters/receivers (each iteration) by using for-loops. The different operations designed for the vectors are the following:

- 2*Nt/2*Nr transmitters/receivers are added (one by one) in a straight line. The positive and negative values of the resulting vectors (T_x/ R_x and -T_x/-R_x) are checked each iteration too.

- 2*Nt/2*Nr transmitters/receivers will be added (one by one) first in one side (right) and then in the another side (left) of the vector T_x /R_x. The positive and negative values of the resulting vectors (T_x/ R_x and -T_x/-R_x) are checked each iteration too.

- 2*Nt and 2*Nr transmitters and receivers are added (one by one) in a straight line respectively. The positive and negative values of the resulting vectors (T_x, R_x) are checked each iteration too.

- 2*Nt and 2*Nr transmitters and receivers will be added (one by one) first in one side (right) and then in the another side (left) of the vector T_x and R_x respectively. The positive and negative values of the resulting vectors (T_x, R_x) are checked each iteration too.

These operations were applied to optimum topologies. Even when the dimensions (length of the transmitters, length of the receivers and length of the antenna) of a previous optimum topology can change, **in general** no optimum arrangements of the vector T_x / R_x will be found by using the operations mentioned before. In this way, it has been proven the needed to optimize for a different value of Nt and Nr every time we want an optimum arrangement of T_x/R_x.

4.2. Dynamic Range=40dB, f_c=5 GHz.

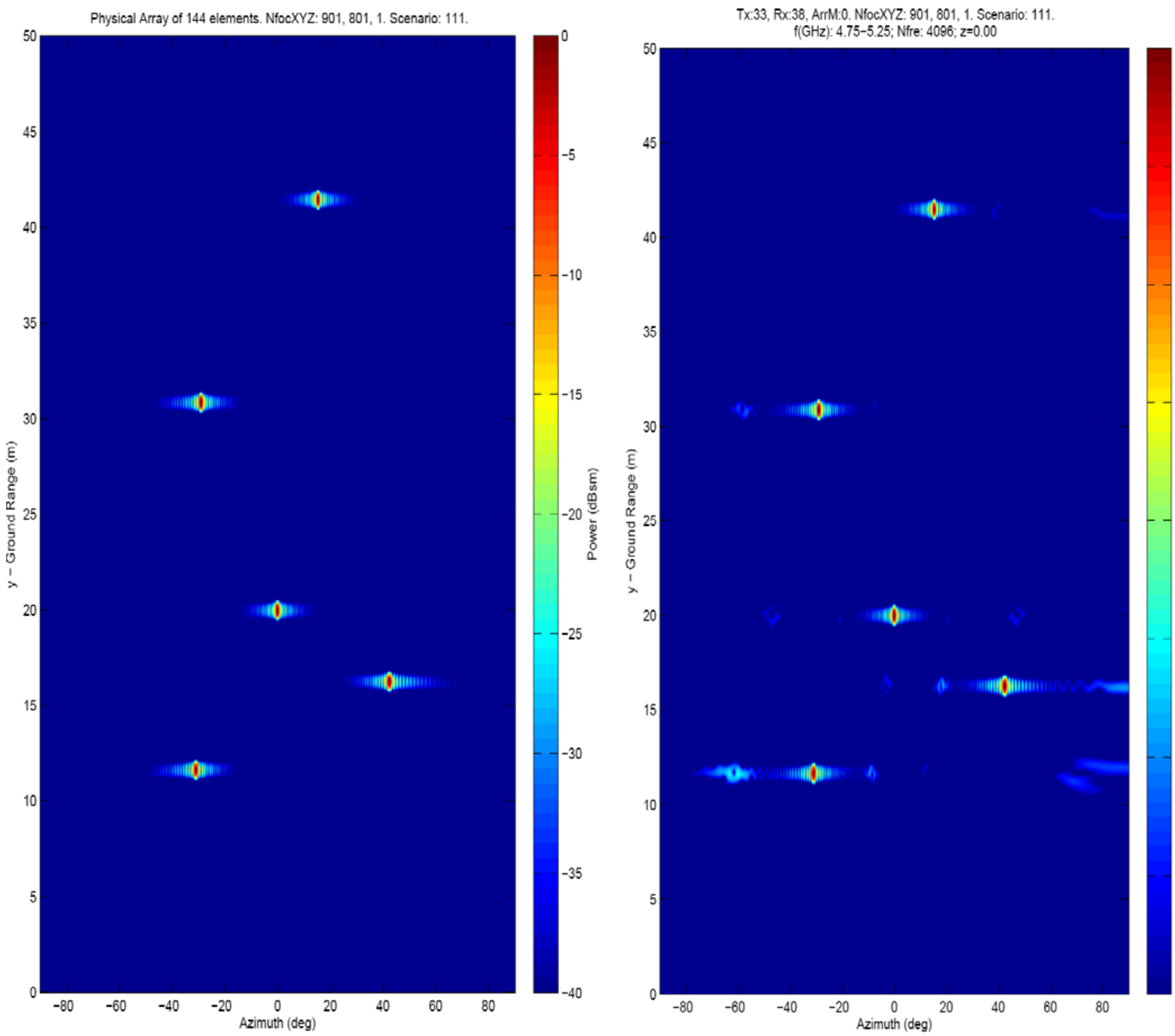


Figure 28.- a. Radar image of the SAR array with 144 elements.

b. Radar image of the MIMO 12x12 array.

Brute Force Algorithm with configuration 1.

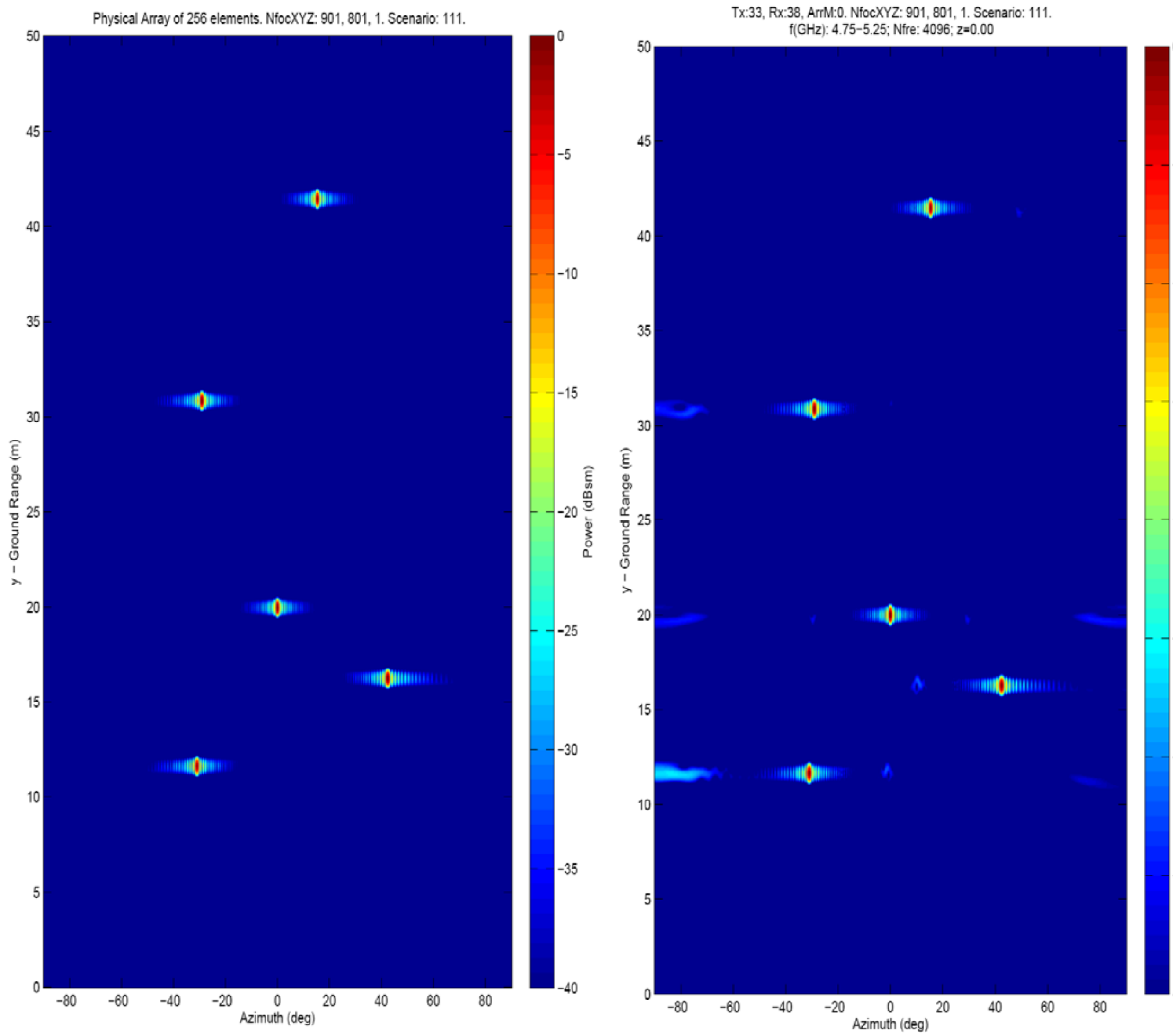


Figure 29.- a. Radar image of the SAR array with 256 elements.

b. Radar image of the MIMO 16x16 array.

Brute Force Algorithm with configuration 1.

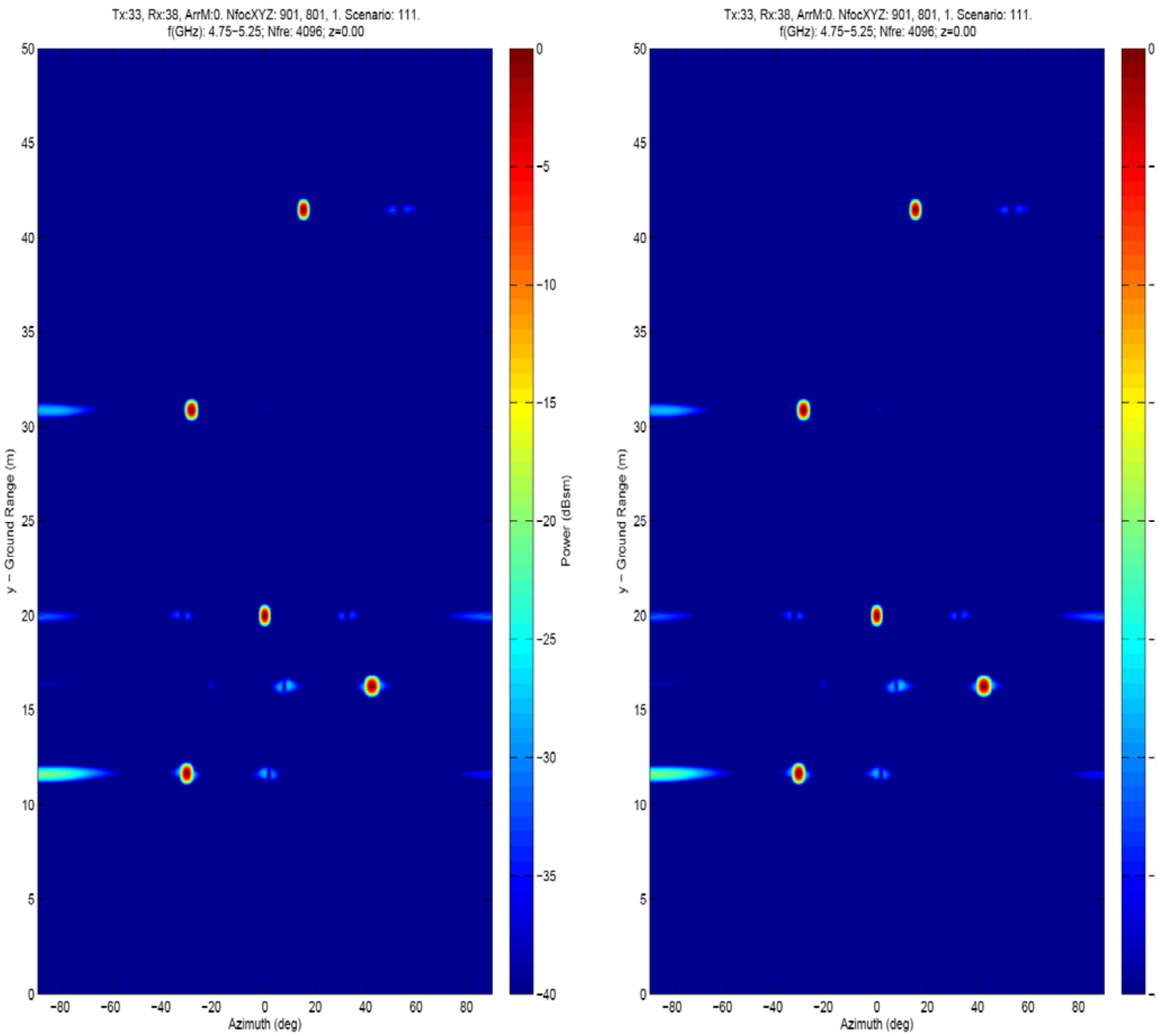


Figure 30.- a. Radar image of the MIMO 16x16 array with BH window with NLLS.

b. Radar image of the MIMO 16x16 array with BH window with GA.

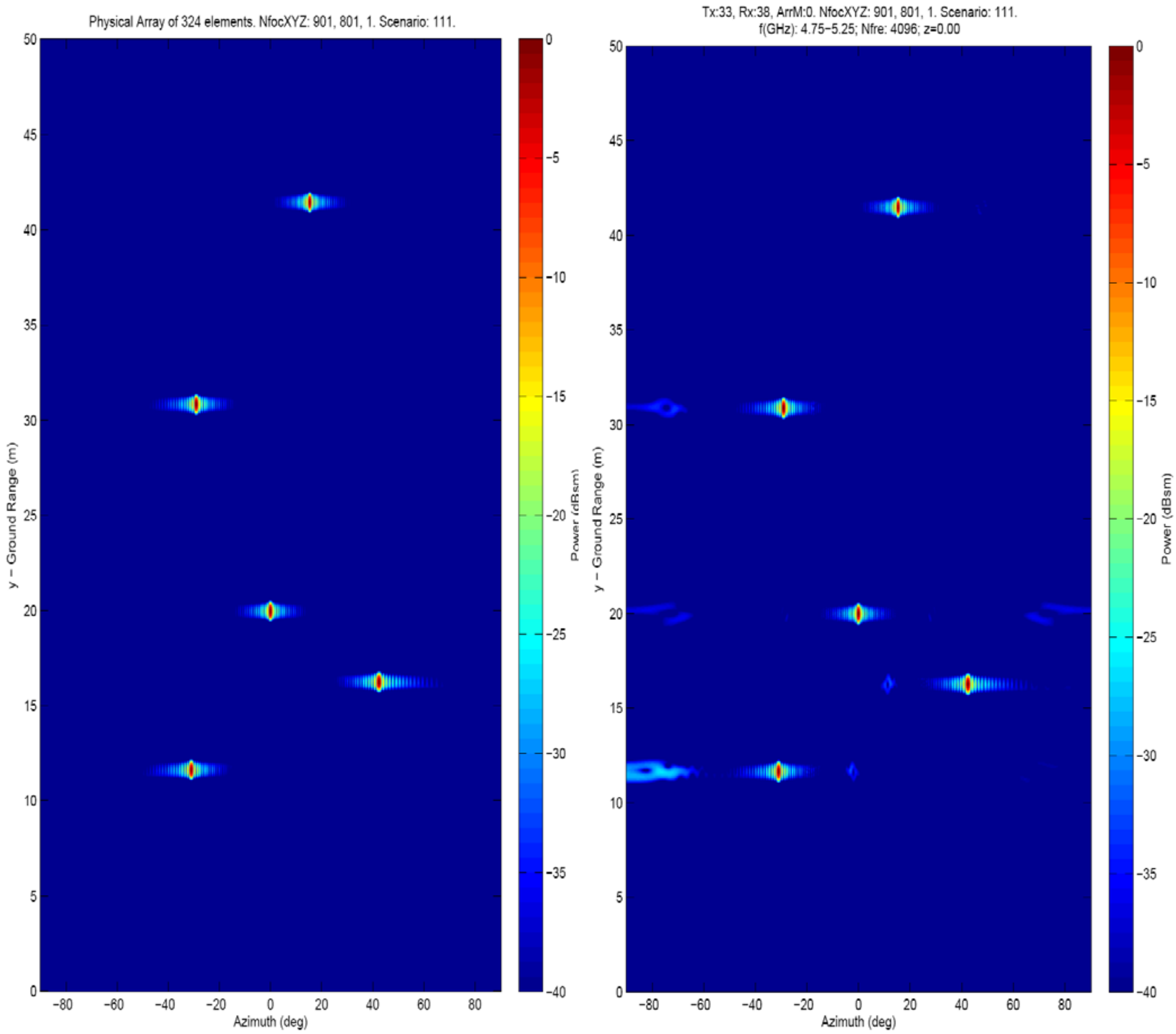


Figure 31.- a. Radar image of the SAR array with 324 elements.

b. Radar image of the MIMO 18x18 array.

Genetic Algorithm with real number codification and configuration 1.

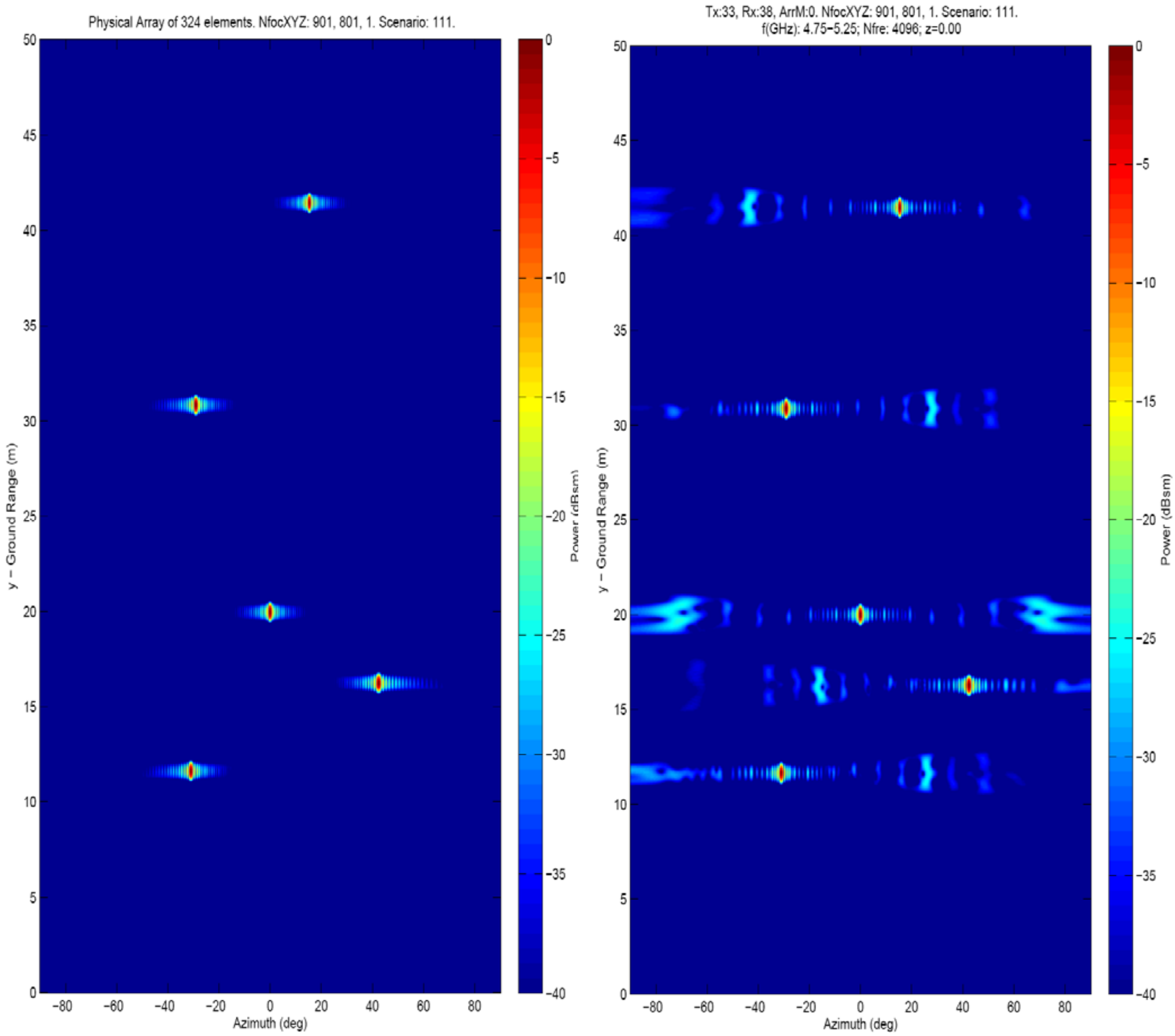


Figure 32.- a. Radar image of the SAR array with 324 elements.

b. Radar image of the MIMO 18x18 array.

Genetic Algorithm bit string codification and configuration 1.

4.3. - Dynamic Range=40dB, f_c=10 GHz.

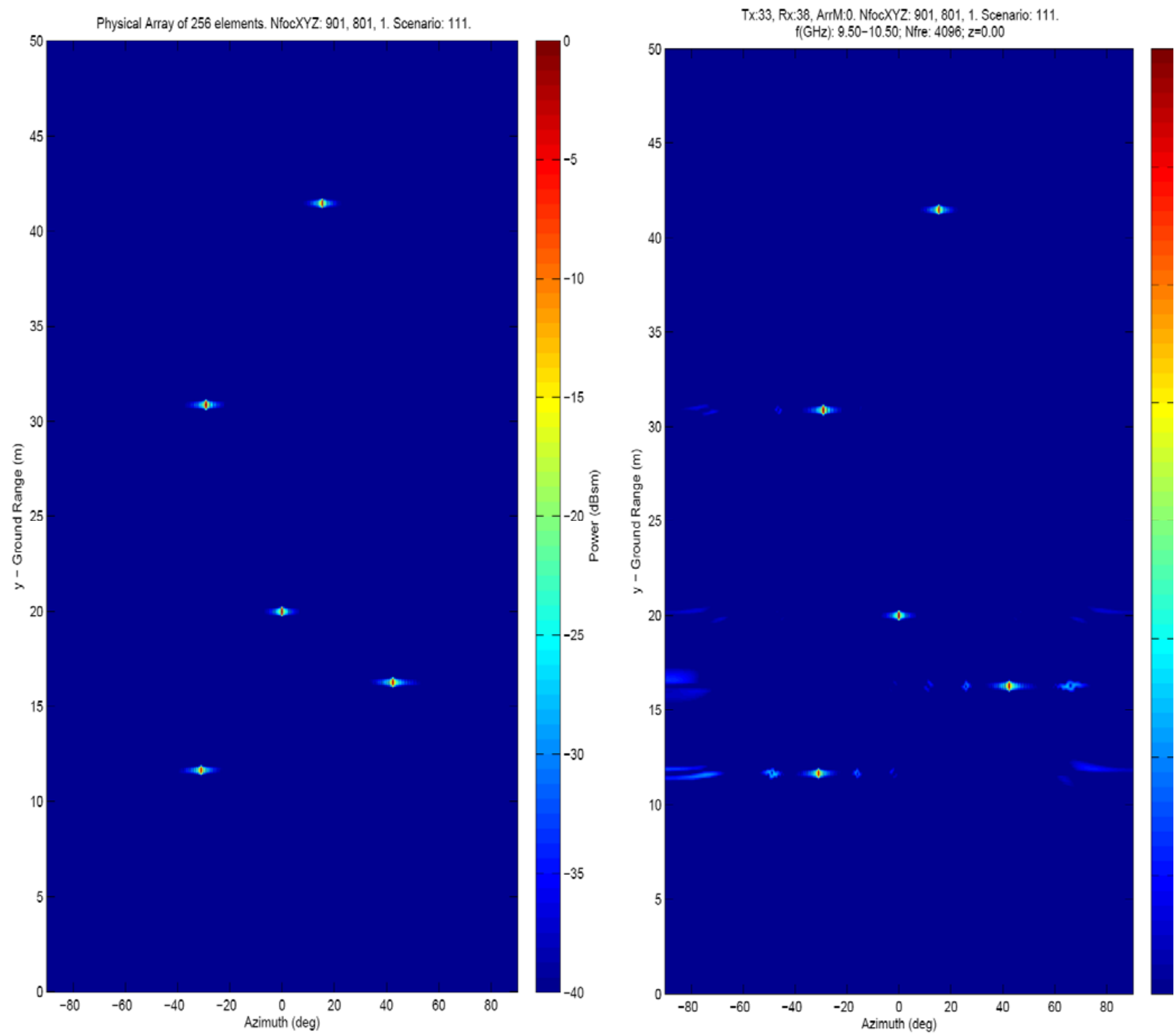


Figure 33.- a. Radar image of the SAR array with 256 elements.

b. Radar image of the MIMO 16x16 array.

Non Linear Least Squares Algorithm.

4.4. - Dynamic Range=70dB, f_c=5 GHz.

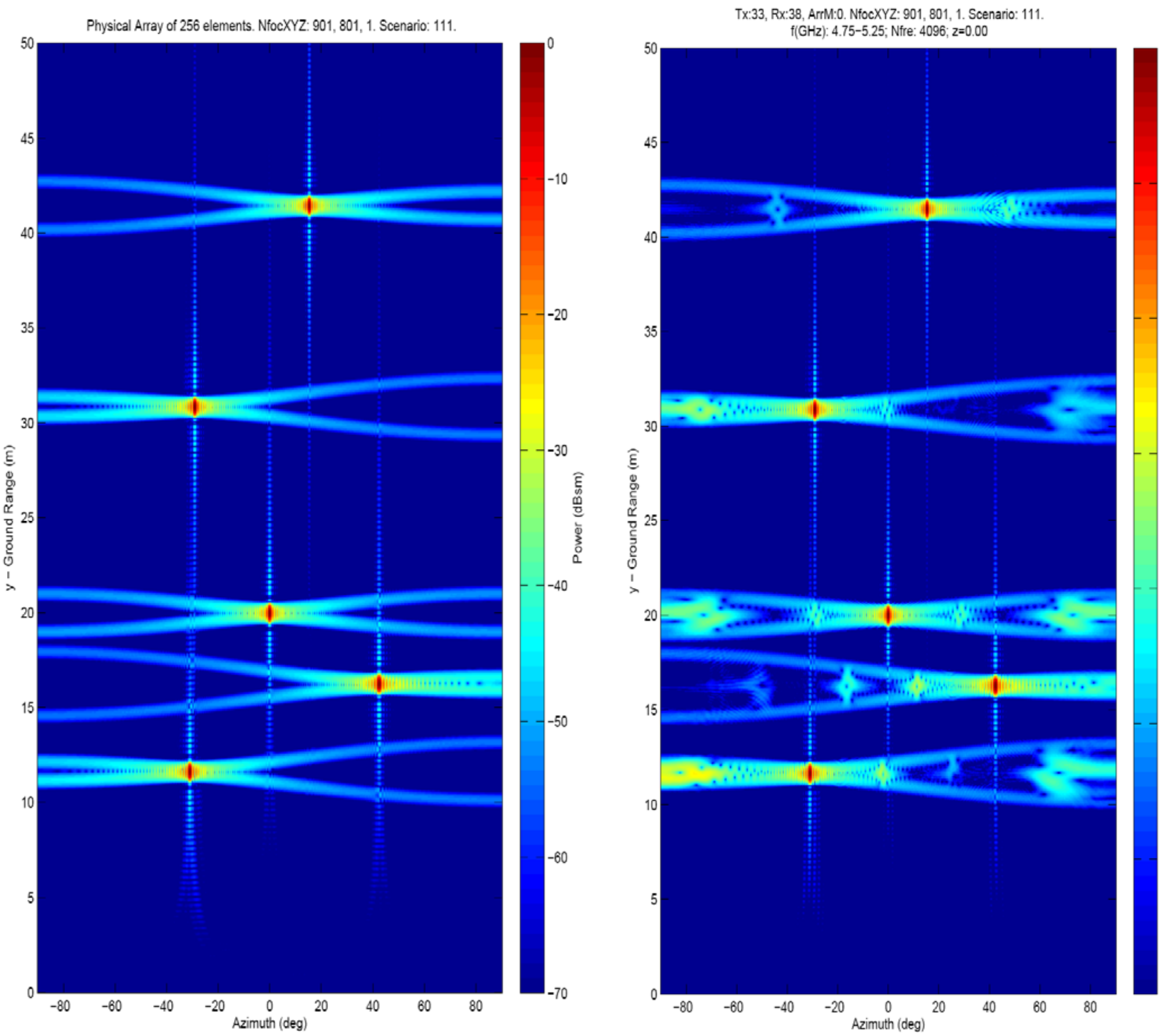


Figure 34.- a. Radar image of the SAR array with 256 elements.

b. Radar image of the MIMO 16x16 array.

Non Linear Least Squares Algorithm.

5. Conclusions.

The proposed optimization techniques of the antenna topology are based on the assumption that the phase centers of the Tx/Rx pairs of the MIMO array must be uniformly distributed and unique. The algorithms commented in this document are able to achieve good results to calculate a good optimum topology. The LLS methods are able to calculate an optimum topology in less than 30 seconds, even though the condition of the aperture length and the effective aperture length is not met. The NLLS method is able to calculate an optimum topology fulfilling all the imposed conditions after applying some calculations to the results it obtains. The drawbacks of this algorithm are the time it takes to get the results and the fact that only optimum topologies with N_t = Nr = (even number) can be found. To reduce the time this method takes an adaptive algorithm to calculate the for-loop-control variables can be performed.

Brute Force algorithms are able to calculate all possible solutions for a reasonable resolution in the search-space. However, if smaller runtimes are required, then a simplification of the search space must be done. Three possible configurations were presented in order to decrease the number of values in the search-space, and proved to be valid.

Some variants of a genetic algorithm were designed according to the configurations calculated with the brute force algorithms to reduce the runtime. They were proved to be a valid tool to obtain optimum topologies with smaller runtimes and similar results than those obtained with BF methods. As an alternative some direct search methods were used. These methods can calculate topologies with a high grade of uniformity in really small runtimes.

The results obtained in figures 28.a, 29.a, 31.a, 32.a, 33.a and 34.a represent the SAR images obtained with the radar for the number of transmitters and receivers indicated in each case. The SAR images are used to compare the results with those obtained with the MIMO radar array. In Figure 30 a Blackman Harris windowing function was applied along x-axis to improve the quality of the images. When this window is applied the similarities between SAR images and MIMO images increase noticeably. To better compare the results between MIMO and SAR images, the results with a dynamic range of 70 dB have been checked (Figure 34).

It has been demonstrated that if an optimum topology (T_{x1}, R_{x1}) with NtxNr elements changes the number of elements (n_1Nt)x(n_2Nr) (with $n_1, n_2=1$, 2, 3, 4, …), then the new topology (T_{x2}, R_{x2}) in general will loss the uniformity and uniqueness condition, and therefore the needed to optimize for every value of Nt and Nr has been proven.

6. References.

[1] A. Martinez Vazquez, J. Fortuny Guasch, "UWB MIMO Radar Arrays For Small Area Surveillance Applications", Proceedings of the 2nd European Conference on Antennas and Propagation, Edimburgh, 11-16 November 2007.

[2] Antenna Theory, Analysis and Design 2nd Edition, Constantine A. Balanis.

[3] Electromagnetic Waves and Antenna, S.J. Orfanidis.

[4] Genetic Algorithm and Direct Search Toolbox 2, The Mathworks, www.mathworks.com.

[5] Radar, signal and image processing techniques for through the wall imaging, Moeness G. Amin, Center for Advanced Communication, Villanova University.

[6] A Genetic Algorithm Tutorial, Darrel Whitley, Computer Science Department, Colorado State University, Fort Collins, CO 80523.

European Commission

EUR 23671 EN – Joint Research Centre – Institute for the Protection and Security of the Citizen
Optimization Techniques for MIMO radar antenna systems
Author: Elias Mendez, Joaquim Fortuny, EC Joint Research Ispra
Luxembourg: Office for Official Publications of the European Communities
2008 – 60 pp. – 21 x 29.7 cm
EUR – Scientific and Technical Research series – ISSN 1018-5593

Abstract
Topology Optimization Technique is a new technique to obtain optimum topologies for MIMO antenna array systems. The target of the optimization is the identification of the optimal arrangement of the transmitters and receivers giving the highest detection performance to obtain radar images as similar as possible to those obtained in SAR techniques. To carry out this task we focused on the concept of the phase center.

How to obtain EU publications

Our priced publications are available from EU Bookshop (http://bookshop.europa.eu), where you can place an order with the sales agent of your choice.

The Publications Office has a worldwide network of sales agents. You can obtain their contact details by sending a fax to (352) 29 29-42758.